Satisfiability, Branch-width and Tseitin Tautologies

Michael Alekhnovich* Alexander A. Razborov†

September 12, 2002

Abstract

For a CNF τ, let $w_b(\tau)$ be the branch-width of its underlying hypergraph. In this paper we design an algorithm for solving SAT in time $n^{O(1)}2^{O(w_b(\tau))}$. This in particular implies a polynomial algorithm for testing satisfiability on instances with tree-width $O(\log n)$.

Our algorithm is a modification of the width based automated theorem prover (WBATP) which is a popular (at least on the theoretical level) heuristic for finding resolution refutations of unsatisfiable CNFs. We show that instead of the exhaustive enumeration of all provable clauses, one can do a better search based on the Robertson-Seymour algorithm for approximating the branch-width of a graph. We call the resulting procedure Branch-Width Based Automated Theorem Prover (BWBATP). As opposed to WBATP, it always produces regular refutations. Perhaps more importantly, the running time of our algorithm is bounded in terms of a clean combinatorial characteristic that can be efficiently approximated, and that the algorithm also produces, within the same time, a satisfying assignment if τ happens to be satisfiable.

In the second part of the paper we investigate the behavior of BWBATP on the well-studied class of Tseitin tautologies. We argue that in this case BWBATP is better than WBATP. Namely, we show that its running time on any Tseitin tautology τ is $|\tau|^{O(1)} \cdot 2^{O(w(\tau \vdash \emptyset))}$, as opposed to the obvious bound $n^{O(w(\tau \vdash \emptyset))}$ provided by WBATP.

*Laboratory for Computer Science, MIT, mishka@theory.lcs.mit.edu

†Institute for Advanced Study, Princeton, US, razborov@ias.edu. On leave from Steklov Mathematical Institute, Moscow, Russia. Supported by The von Neumann Fund.

This in particular implies that Resolution is automatizable on those Tseitin tautologies for which we know the relation $w(\tau \vdash \emptyset) \leq O(\log S(\tau))$. We identify one such subclass and prove partial results toward establishing this relation for larger classes of graphs.

1. Introduction

Satisfiability is probably one of the most important NP-complete problems for which various ad hoc heuristics have been designed. The reason why these heuristics are so popular comes from two sources: first, SAT is the most canonical NP-complete problem; second, many practical applications do require either a satisfying assignment or a proof of unsatisfiability, and are not satisfied with approximate solutions, thereby excluding the possibility to use the powerful tools from the theory of approximation algorithms.

Much attention in Computer Science has been recently devoted to theoretical study and justification of these ad-hoc procedures. It seems that despite its (commonly believed) worst-case hardness, there are still many interesting issues about Satisfiability, where a progress can be made.

In this paper we investigate the hardness of SAT from the perspective of *branch-width* of the underlying hypergraph (the latter measure is equivalent up to a constant factor to more popular *tree-width* of a graph). There has been a huge amount of prior related work done in this direction, we will not even try to give all the references and confine ourselves only to the most relevant ones. The idea to decompose a graph into smaller connected components in order to solve an associated hard problem by divide-n-conquer techniques originally goes back to Lipton and Tarjan [LT80]. Using a consequence of their celebrated planar separator theorem [LT79] they constructed many algorithms for solving NP-complete problems on planar graphs in time $2^{O(\sqrt{n})}$, which is much faster than the brute-force search. Since then, much theoretical research has been done in the direction of improving the decomposition search heuristics, as well as applying these techniques for a wider class of problems (see for example [Bod93] for a survey).

As to the SAT problem itself, there was a work by Khanna and Motwani [KM96], which, besides other things, considers a CNF as a bipartite graph and deals with the problem on instances of constant tree-width. However, their goal is to approximate the number of satisfied clauses rather than to solve it exactly. Similary, in the work [CMR01] the hardness of SAT was studied in terms of so-called clique-width of a graph. This result is more general than ours in that it can be applied to a wider class of graphs, but it gives substantially weaker upper bounds (only $n^{O(1)}f(w)$, where f is an unspecified recursive function of the clique-width w). Finally, some recent empirical SAT heuristics exploit *tree decomposition* to find a satisfying assignment (see, for example, [AM01]).

In this paper we apply Robertson-Seymour branch-width approximating algorithm (generalized to hypergraphs) for solving satisfiability and automatic proof search in the system of Resolution. Namely, we construct the heuristic which we call *Branch-Width Based Automated Theorem Prover* (BWBATP for short) that given a CNF τ checks its satisfiability in time $n^{O(1)}2^{O(w_b(\tau))}$, where $w_b(\tau)$ is the branch-width of the underlying hypergraph of τ. If the input is not satisfiable, it produces a regular resolution refutation of nearly the same size and of width $O(w_b(\tau))$.

The second part of the paper is dedicated to the more theoretical issue of automatizability of Resolution. In brief, a propositional proof system is automatizable if the proof of any tautology can be efficiently found in time polynomial in the size of its shortest proof[1]. It was shown by Alekhnovich and Razborov [AR01b] that modulo a plausible hardness assumption from the theory of parameterized complexity, Resolution is not automatizable. The examples used in that paper, however, look somewhat ad hoc and isolated. Thus, it is natural to wonder (especially for someone who is in general critical about the worst-case complexity) to which extent these examples are relevant in practice.

Our main (informal) goal in this direction is to address the question by identifying some natural classes of tautologies on which Resolution is automatizable by BWBATP in the above sense, i.e, the prover finds refutations within time that is close to optimal. Our main target is the class of Tseitin tautologies. However, the concepts and some of the results found on this way might have a broader applicability. In order to formulate them, let us pinpoint the two problems that naturally arise in the construction of Resolution proof search heuristics by the following structural definitions.

[1]It is good to notice that the notion of automatizability is in a sense "orthogonal" to the notion of efficiency of the proof system. In particular, if optimal proofs tend to have larger sizes then it might be *easier* to find them in the spirit of this definition since the algorithm is allowed more running time.

Say that a class $\mathcal{C}$ of unsatisfiable CNFs is *width automatizable* if there exists a deterministic algorithm that for every $\tau \in \mathcal{C}$ constructs its resolution refutation of width $O(w(\tau \vdash \emptyset))$ in time $|\tau|^{O(1)} \cdot 2^{O(w(\tau \vdash \emptyset))}$. Say that $\mathcal{C}$ is *smooth* if for $\tau \in \mathcal{C}$ we have $w(\tau \vdash \emptyset) \leq K_{\mathcal{C}} \log S(\tau)$ for some absolute constant $K_{\mathcal{C}} > 0$. Thus, every class $\mathcal{C}$ that is at the same time width automatizable and smooth is automatizable. It is known that in general Resolution is neither width-automatizable (modulo a plausible assumption, [AR01b]) nor smooth (see Bonet, Galesi [BG99] and Atserias, Bonet [AB02]). We address these issues separately for more narrow classes.

Concerning the first one, our main concrete result (Theorem 2.8) says that the class of all Tseitin tautologies is width automatizable. We prove this by showing that in this case $w_b(\tau) = \theta(w(\tau \vdash \emptyset))$.

Our contributions toward the second issue (smoothness) are by far more limited. Let the *cyclicity* $\ell(G)$ of a graph G be the minimum ℓ for which the edges of G can be covered by cycles of length at most ℓ in such a way that every edge belongs to at most ℓ cycles. We prove (Theorem 2.10) that the class of Tseitin tautologies based on graphs of bounded cyclicity is smooth. In combination with Theorem 2.8, this implies that this class is automatizable with respect to Resolution. The class of graphs of bounded cyclicity in particular encompasses the class of all planar graphs with bounded degree of faces, of which the grid (considered in the original paper [Tse68]) is the typical example. Even if we drop the second restriction in the definition of cyclicity (i.e., do not restrict the number of cycles an edge may belong to), we still can show that the corresponding class of Tseitin tautologies is *quasi-smooth*, that is, that $w(\tau \vdash \emptyset) \leq \log S(\tau)^{O(1)}$ (Theorem 2.12). In particular, this class is quasi-automatizable. We also include one weak partial result toward showing (quasi)-smoothness of the class of all Tseitin tautologies (Theorem 2.13).

The paper is organized as follows. In Section 2 we give the necessary background and formulate our main results. We construct the main algorithm and investigate its performance in Section 3. Section 4 contains the constructions and proofs related to width automatizability of Tseitin tautologies whereas Section 5 is devoted to the proofs of our smoothness results. Finally, the paper is concluded in Section 6 with a few open problems.

2. Preliminaries and Results

Let x be a Boolean variable, i.e. a variable that ranges over the set $\{0, 1\}$. A *literal* of x is either x (denoted sometimes as x^1) or $\bar{x}$ (denoted sometimes as x^0). A *clause* is a disjunction of literals. A *CNF* is a conjunction of pairwise different clauses, often identified with the set consisting of these clauses.

For a clause C, $Vars(C)$ is the set of variables appearing in C. For a CNF τ, let $Vars(\tau) \stackrel{\text{def}}{=} \bigcup\{Vars(C) \mid C \in \tau\}$.

An *assignment to the set of variables* V is a mapping $\alpha : V \to \{0, 1\}$. A *restriction* is a mapping $\rho : V \to \{0, 1, \star\}$. The *restriction of a clause* C *or CNF* τ *by* ρ, denoted by $C|_\rho$ $[\tau|_\rho]$ is the clause [CNF] obtained from C [τ, respectively] by setting the value of each $x \in \rho^{-1}(\{0, 1\})$ to $\rho(x)$, and leaving each $x \in \rho^{-1}(\star)$ as a variable. Say that a CNF τ *semantically implies a clause* C (denoted $\tau \models C$) if every assignment to $Vars(\tau)$ satisfying all clauses in τ, sends C to 1.

One of the simplest and the most widely studied propositional proof systems is *Resolution* which operates with clauses and has one rule of inference called *resolution rule*:

$$\frac{A \vee x \qquad B \vee \bar{x}}{A \vee B}.$$

We say that the variable x is *resolved* in this instance of the resolution rule. A resolution proof is *regular* if along each path every variable is resolved at most once. (Regular) Resolution is *implicationally sound and complete*, that is a (regular) resolution proof of C from τ exists if and only if $\tau \models C$. A *resolution refutation* of a CNF τ is a resolution proof of the empty clause from the clauses appearing in τ.

The *size* of a resolution proof is the overall number of clauses in it. For an unsatisfiable CNF τ, $S(\tau)$ is the minimal size of its resolution refutation. The *width* $w(C)$ *of a clause* C is the number of literals in C. The *width* $w(\tau)$ *of a set of clauses* τ (in particular, the width of a resolution proof) is the maximal width of a clause appearing in this set. For an unsatisfiable CNF τ, $w(\tau \vdash \emptyset)$ will denote the minimal width of its resolution refutation. Let also $n(\tau)$ be the overall number of distinct variables appearing in τ, and $|\tau|$ be the overall number of occurrences of variables in τ, i.e., $|\tau| \stackrel{\text{def}}{=} \sum_{C \in \tau} w(C)$.

For n, a non-negative integer let $[n] \stackrel{\text{def}}{=} \{1, 2, \ldots, n\}$.

A *hypergraph* is a pair $\mathcal{H} = (V, \mathcal{E})$, where V is a finite set of *vertices*, and $\mathcal{E}$ is a multiset whose elements (called *hyperedges*) are subsets of V

(thus, multiple edges are allowed). The hypergraph $(V, \mathcal{E})$ is a *graph* if all edges $E \in \mathcal{E}$ have cardinality 2, in which case they will be denoted by small letters e. The *star* of a vertex v is $S_{\mathcal{H}}(v) \stackrel{\text{def}}{=} \{E \in \mathcal{E} \mid v \in E\}$, and its *degree* is $\deg_{\mathcal{H}}(v) \stackrel{\text{def}}{=} |S_{\mathcal{H}}(v)|$. The *dual hypergraph* $\mathcal{H}^*$ is the hypergraph $\{S_{\mathcal{H}}(v) \mid v \in V\}$ on $\mathcal{E}$. $r(\mathcal{H}) \stackrel{\text{def}}{=} \max_{E\in\mathcal{E}} |E|$ is the maximum size of a hyperedge.

The following definition of the branch-width of a hypergraph is essentially the same as in [RS91]; the only (cosmetic) change we have made to it is to consider (binary) rooted trees instead of (ternary) unrooted ones.

Definition 2.1 A *branch-decomposition* of a hypergraph $\mathcal{H} = (V, \mathcal{E})$ is a pair (T, θ), where T is a binary rooted tree and θ is a bijection between $\mathcal{E}$ and the set of leaves of T. For an arbitrary node t of T, let $\mathcal{E}(t) = \mathcal{E}_\theta(t) \stackrel{\text{def}}{=} \{E \in \mathcal{E} \mid \theta(E) \text{ is a descendant of } t\}$, and let also $\mathcal{E}^\perp(t) \stackrel{\text{def}}{=} \mathcal{E} \setminus \mathcal{E}(t)$. Let $V(t) \stackrel{\text{def}}{=} \bigcup \{E \mid E \in \mathcal{E}(t)\}$ and, respectively, $V^\perp(t) \stackrel{\text{def}}{=} \bigcup \left\{E \,\middle|\, E \in \mathcal{E}^\perp(t)\right\}$. The *cut of* t is $\mathrm{Cut}(t) \stackrel{\text{def}}{=} V(t) \cap V^\perp(t)$. The *order of* t is the cardinality of $\mathrm{Cut}(t)$. The *width* $w(T, \theta)$ *of the branch-decomposition* (T, θ) is the maximum order of nodes in T, and the *branch-width of* $\mathcal{H}$, denoted by $w_b(\mathcal{H})$ is the minimum width of all branch-decompositions of $\mathcal{H}$.

Given a CNF $\tau = C_1 \wedge \ldots \wedge C_m$, we associate with it the hypergraph

$$\mathcal{H}_\tau \stackrel{\text{def}}{=} \{Vars(C_1), \ldots, Vars(C_m)\}$$

on $Vars(\tau)$. Let the *branch-width* $w_b(\tau)$ *of* τ be the branch-width of this hypergraph: $w_b(\tau) \stackrel{\text{def}}{=} w_b(\mathcal{H}_\tau)$.

Remark 1 Branch-width is known to be equivalent (up to a multiplicative constant) to the more popular notion of tree-width [RS91]. However, we prefer to state our results in terms of the former measure because we will use a generalization of Robertson-Seymour algorithm approximating branch-width. Also, concepts and constructions associated with branch-width look by far more natural in our specific context.

The following result basically says that branch-decompositions can be used for constructing efficient resolution refutations.

Theorem 2.2 *There exists a deterministic algorithm which on every CNF* τ *runs in time polynomial in* $|\tau| + 2^{w_b(\tau)}$ *and:*

1. *returns its regular resolution refutation of width $O(w_b(\tau))$ if τ is unsatisfiable;*

2. *returns a satisfying assignment if τ is satisfiable.*

Recall the general definition of automatizability from [BPR00]. We give it here only for the special cases of Resolution; on the other hand, it is presented for arbitrary classes of unsatisfiable CNFs (as opposed to the class of *all* unsatisfiable CNFs in [BPR00]).

Definition 2.3 A class $\mathcal{C}$ of unsatisfiable CNFs is (*quasi-*)*automatizable* (with respect to Resolution) if there exists a deterministic *automatizing algorithm* that, given an unsatisfiable CNF $\tau \in \mathcal{C}$, returns its resolution refutation in time which is (quasi-)polynomial in $|\tau| + S(\tau)$.

Note that the size of the resolution refutation constructed by any automatizing algorithm should also necessarily be (quasi-)polynomial in $|\tau| + S(\tau)$.

Now we introduce along the same lines the following concept, technical but very useful.

Definition 2.4 A class $\mathcal{C}$ of unsatisfiable CNFs is *width-automatizable* if there exists a deterministic algorithm that, given an unsatisfiable CNF $\tau \in \mathcal{C}$, returns its resolution refutation of width $O(w(\tau \vdash \emptyset))$ in time which is polynomial in $|\tau| + 2^{w(\tau \vdash \emptyset)}$.

Note that the procedure exhaustively generating all provable clauses of width $\leq w$, for $w = 1, 2, \ldots, n(\tau)$ (*Width Based Automated Theorem Prover*) accomplishes this task within time $|\tau|^{O(w(\tau \vdash \emptyset))}$, that is, *the class of all unsatisfiable CNFs is quasi-width-automatizable.*

Ben-Sasson and Wigderson proved in [BW01] that $w(\tau \vdash \emptyset) \leq O(\log S_T(\tau))$, where $S_T(\tau)$ is the minimum size of a *tree-like* resolution refutation. Motivated by this result, we introduce the following concept.

Definition 2.5 A class $\mathcal{C}$ of unsatisfiable CNFs is *smooth* [*quasi-smooth*] if there exists an absolute constant $K_{\mathcal{C}} > 0$ such that $w(\tau \vdash \emptyset) \leq K_{\mathcal{C}} \log S(\tau)$ [$w(\tau \vdash \emptyset) \leq (\log S(\tau))^{K_{\mathcal{C}}}$, respectively] for all $\tau \in \mathcal{C}$.

The following immediately follows from definitions.

Fact 1 *Every class of unsatisfiable CNFs that is at the same time width automatizable and smooth is automatizable. Every quasi-smooth class is quasi-automatizable.*

Theorem 2.2 in particular implies the bound $w(\tau \vdash \emptyset) \leq O(w_b(\tau))$. We now identify one situation in which this bound is tight.

Definition 2.6 Let $G = (V(G), E(G))$ be an (ordinary) connected graph, and $\sigma : V(G) \to \{0, 1\}$ be an *odd-weight function*, i.e., such that $\bigoplus_{v \in V(G)} \sigma(v) = 1$. Assign a distinct variable x_e to each edge $e \in E(G)$. For $v \in V(G)$ and $\epsilon \in \{0, 1\}$, let

$$PARITY_{v,\epsilon} \stackrel{\text{def}}{=} \bigwedge \left\{ \bigvee_{e \ni v} x_e^{a(e)} \,\middle|\, \bigoplus_{e \ni v} (a(e) \oplus 1) \neq \epsilon \right\}$$

be the straightforward CNF representation of the Boolean function

$$\bigoplus_{e \ni v} x_e = \epsilon. \tag{1}$$

The *Tseitin tautology* of G and σ is the unsatisfiable CNF given by

$$T(G, \sigma) \stackrel{\text{def}}{=} \bigwedge_{v \in V} PARITY_{v,\sigma(v)}.$$

Remark 2 In the proof-complexity studies one often assumes that the vertex degrees of G are bounded, and that G has certain expansion properties. None of these two assumptions is either needed or relevant to our purposes.

Remark 3 The hypergraph $\mathcal{H}_{T(G,\sigma)}$ associated with the Tseitin tautology $T(G, \sigma)$ is just the dual hypergraph G^*, with the difference that every hyperedge of size r repeats itself 2^{r-1} times.

Theorem 2.7 $w_b(T(G, \sigma)) = \theta(w(T(G, \sigma) \vdash \emptyset))$.

Comparing this with Theorem 2.2, we immediately get our main result concerning width automatizability.

Theorem 2.8 *The class of all Tseitin tautologies is width-automatizable.*

Let us now turn to our second issue, smoothness.

Definition 2.9 For an ordinary graph G, define its *cyclicity* as the minimal ℓ for which the edges of G can be covered by cycles in such a way that every cycle has length at most ℓ and every edge belongs to at most ℓ cycles.

Theorem 2.10

$$w(T(G, \sigma) \vdash \emptyset) \leq \ell(G)^{O(1)} \cdot \log S(T(G, \sigma)).$$

In particular, the class of all Tseitin tautologies $T(G, \sigma)$ for which $\ell(G)$ is bounded by some absolute constant is smooth.

Theorems 2.8 and 2.10 imply

Theorem 2.11 *The class of all Tseitin tautologies $T(G, \sigma)$ for which $\ell(G)$ is bounded by some absolute constant is automatizable.*

We remark that this class of graphs in particular encompasses planar graphs in which all faces have bounded degree.

Let $\tilde{\ell}(G)$ be the minimal ℓ for which the edges of G can be covered by cycles of length $\leq \ell$ (with no restrictions on the multiplicity of this cover). We also have the following:

Theorem 2.12

$$w(T(G, \sigma) \vdash \emptyset) \leq (\tilde{\ell}(G) \log S(T(G, \sigma)))^{O(\tilde{\ell}^2(G))}.$$

In particular, the class of all Tseitin tautologies $T(G, \sigma)$ for which $\tilde{\ell}(G)$ is bounded by some absolute constant is quasi-smooth and, hence, quasi-automatizable.

Finally, let us mention a simple partial result toward showing smoothness of the class of all Tseitin tautologies:

Theorem 2.13 $w(T(G, \sigma) \vdash \emptyset) \leq O(\log S(T(G, \sigma)) + |V(G)|)$.

3. Branch-Width Based Automated Theorem Prover

In this section we prove Theorem 2.2. The algorithm we construct will proceed in two fairly independent stages. At the first stage we show how to find a nearly optimal branch-decomposition (Lemma 3.1); this will be done by a fairly straightforward adaptation of the Robertson-Seymour algorithm to hypergraphs. At the second stage we show how to use any narrow branch-decomposition for constructing efficient resolution refutations (Lemma 3.2).

Lemma 3.1 *There is a deterministic algorithm which for every hypergraph $\mathcal{H} = (V, \mathcal{E})$ outputs its branch-decomposition of width $O(w_b(\mathcal{H}))$ within time $(|V| + |\mathcal{E}|)^{O(1)} \cdot 2^{O(w_b(\mathcal{H}))}$.*

Proof of Lemma 3.1. Let us first observe that the vertices v with $\deg_{\mathcal{H}}(v) \leq 1$ do not contribute to the branch-width and can be immediately deleted. Thus, we may assume w.l.o.g. that $\deg_{\mathcal{H}}(v) \geq 2$ for all $v \in V$. This in particular implies (by looking at the leaf corresponding to the hyperedge of maximal size) that

$$w_b(\mathcal{H}) \geq r(\mathcal{H}). \tag{2}$$

Define the oriented graph $G = (V, F)$ on the same vertex set V by letting

$$F \stackrel{\text{def}}{=} \{(v, v') \mid \exists E \in \mathcal{E}(\{v, v'\} \subseteq E)\}.$$

Our basic idea is to apply to G the Robertson-Seymour algorithm for approximating the branch-width of ordinary graphs [RS95]. This reduction from $\mathcal{H}$ to G is not absolutely straightforward (since we have to be careful not to cut the original hyperedges), so we prefer to give an independent description in a language convenient for our further purposes. Also, we do not attempt to optimize the multiplicative constant which will make the algorithm conceptually simpler.

Let a *partial branch-decomposition* be defined in the same way as a (total) branch-decomposition, with the difference that we require the function θ to be only surjective and not necessarily one-to-one. A partial branch-decomposition $(\hat{T}, \hat{\theta})$ is a *refinement* of another partial branch-decomposition (T, θ) if T is a sub-tree of $\hat{T}$ and for every node t of T, $\mathcal{E}_\theta(t) = \mathcal{E}_{\hat{\theta}}(t)$. The algorithm tries to construct a (total) branch-decomposition of width $\leq w$, when the parameter w is incremented by one at a time. Given a particular value of w, the algorithm begins with the trivial partial branch-decomposition (i.e., when T is a single-node tree) and keeps refining it in such a way that the width never gets above w, until it either arrives at a complete branch-decomposition or gets stuck. The crucial fact will be that

begin
(1) **if** $|V| < 4r(\mathcal{H})$ **return** an arbitrary branch-decomposition **else**
(2) **for** $w = 4r(\mathcal{H}), 4r(\mathcal{H}) + 1, \ldots$ **do**
  begin
    $T := \{root\}$ (a single-node tree);
    $\theta :=$ the function which sends every $E \in \mathcal{E}$ to *root*;
(3)   **if** θ is one-to-one **then return** (T, θ) **else**
    $t :=$ an arbitrary leaf of T with $|\theta^{-1}(t)| > 1$;
    $Z := \mathrm{Cut}(t)$;
(4)   **if** $|Z| \leq w - r(\mathcal{H})$ **do**
    begin
      branch T at t into two new leaves t_1, t_2;
      for $E \in \theta^{-1}(t)$ **do** $\theta(E) := \begin{cases} t_1 \text{ if } E = E_1 \\ t_2 \text{ otherwise} \end{cases}$, where E_1 is an arbitrary fixed element of $\theta^{-1}(t)$;
      go to (3)
    end
(5)   **if** (there exists a partition $V = C \,\dot\cup\, V_1 \,\dot\cup\, V_2$ with $|C| \leq \frac{1}{6}|Z|$ such that $F \cap (V_1 \times V_2) = \emptyset$ and $|V_1 \cap Z|,\ |V_2 \cap Z| \geq \frac{1}{6}|Z|$) **do**
    begin
(6)   pick any such partition;
      comment *note that in particular every hyperedge* $E \in \theta^{-1}(t)$ *either does not intersect* V_1 *or does not intersect* V_2;
      branch T at t into two new leaves t_1, t_2;
      for $E \in \theta^{-1}(t)$ **do** $\theta(E) := t_{3-\nu}$, where ν is such that $E \cap V_\nu = \emptyset$;
      comment *if* $E \cap V_1 = E \cap V_2 = \emptyset$*,* ν *is chosen arbitrarily*;
      go to (3)
    end
    go to (2)
    comment *in all other cases the algorithm gets stuck and returns to the operator* **(2)** *incrementing the value of* w *by* 1*. In this case we will call the current value of* Z **an obstacle**;
  end
end

Figure 1: Constructing branch-width decomposition in a hypergraph $\mathcal{H}$

in the latter case the algorithm will find “an obstacle” that will guarantee the inequality $w_b(\mathcal{H}) \geq \Omega(w)$. Formally the algorithm is shown on page 11.

Let us analyze this algorithm. First observe that it maintains the property $w(T, \theta) \leq w$. Indeed, if the refinement is done according to the operator **(4)**, then for $\nu \in \{1, 2\}$ we have $\mathrm{Cut}(t_\nu) \subseteq Z \cup E_1$ which implies $|\mathrm{Cut}(t_\nu)| \leq (w - r(\mathcal{H})) + r(\mathcal{H}) \leq w$. If the refinement proceeds according to the operator **(5)**, then $\mathrm{Cut}(t_1) \subseteq C \cup (V_1 \cap (V_2 \cup Z)) = C \cup (V_1 \cap Z)$, and $|C| + |V_1 \cap Z| \leq \frac{1}{6}|Z| + (|Z| - |V_2 \cap Z|) \leq |Z| \leq w$ by the inductive assumption. A similar argument shows that $|\mathrm{Cut}(t_2)| \leq w$.

Next, note that all steps of this algorithm can be performed within time polynomial in $|V| + |\mathcal{E}|$, except for the operators **(5)**, **(6)**. But these operators can be implemented within time $2^{O(w)}$ simply by searching exhaustively through all disjoint pairs of subsets $Z_1, Z_2 \subseteq Z$ such that $|Z_1|, |Z_2| \geq \frac{1}{6}|Z|$ and there is no edge in G between them. For every such pair (Z_1, Z_2) we find the minimum vertex cut C separating Z_1 and Z_2, and return this cut in the operator **(6)** if $|C| \leq \frac{1}{6}|Z|$. Thus, the overall running time of the algorithm is bounded by $(|V| + |\mathcal{E}|)^{O(1)} \cdot 2^{O(w)}$, where w is the width of the branch-decomposition the algorithm finally returns.

We are only left to prove that $w \leq O(w_b(\mathcal{H}))$. If the algorithm returns the branch-decomposition according to the operator **(1)**, or if $w = 4r(\mathcal{H})$, this follows from (2). Otherwise, let Z be the obstacle that made the algorithm to abort at the $(w-1)$st stage. Note that, due to the presence of the operator **(4)**, $|Z| \geq w - r(\mathcal{H})$.

Let (T, θ) be the total branch-decomposition[2] of $\mathcal{H}$ that has width $w_b(\mathcal{H})$. For a node t of T, let $Z(t) \stackrel{\mathrm{def}}{=} V(t) \cap Z$ and $Z^\perp(t) \stackrel{\mathrm{def}}{=} V^\perp(t) \cap Z$. Clearly, $Z(root) = Z$ and for any leaf t, $|Z(t)| \leq r(\mathcal{H}) < \frac{1}{3}(w - r(\mathcal{H})) \leq \frac{1}{3}|Z|$. Therefore, there exists an internal node t such that $|Z(t)| > \frac{1}{3}|Z|$, whereas $|Z(t_1)| \leq \frac{1}{3}|Z|$ and $|Z(t_2)| \leq \frac{1}{3}|Z|$ for both its children t_1 and t_2. In particular, $|Z(t)| \leq |Z(t_1)| + |Z(t_2)| \leq \frac{2}{3}|Z|$ which implies $|Z^\perp(t)| \geq \frac{1}{3}|Z|$ (since $Z(t) \cup Z^\perp(t) = Z$). We let $C := \mathrm{Cut}(t)$. Then, after deleting C, $Z(t) \setminus C$ and $Z^\perp(t) \setminus C$ become disconnected in the graph G. Since Z is an obstacle, we must have $|C| > \frac{1}{6}|Z|$ which implies the required bound: $w_b(\mathcal{H}) \geq |C| > \frac{1}{6}|Z| \geq \frac{1}{6}(w - r(\mathcal{H})) \geq \frac{1}{6}(w - \frac{w}{4}) = \frac{w}{8}$. Lemma 3.1 is completely proved.∎

[2] (T, θ) does not have anything to do with the partial branch-decomposition the algorithm has constructed at the moment of abortion, and in fact the latter will not be used in the rest of the argument at all

Now we show how to convert branch-decompositions into resolution refutations.

Lemma 3.2 *There exists a deterministic algorithm which for every unsatisfiable CNF τ and every branch-decomposition (T, θ) of the hypergraph $\mathcal{H}_\tau$ returns a regular resolution refutation P of τ in time polynomial in $|\tau|$ and $2^{w(T,\theta)}$. Moreover, P has width $O(w(T, \theta))$.*

Proof of Lemma 3.2. We need the following well-known construction.

Definition 3.3 For a set of clauses $\mathcal{C}$, let $\ker(\mathcal{C})$ be its subset that consists of minimal (with respect to inclusion) clauses. Let

$$\mathcal{C}^x \stackrel{\text{def}}{=} \ker\left[\{C \vee D \mid C \vee x,\ D \vee \bar{x} \in \mathcal{C}\} \cup \{C \in \mathcal{C} \mid x \notin Vars(\mathcal{C})\}\right]$$

be the set of clauses obtained from $\mathcal{C}$ by all possible resolutions upon the variable x. It is not hard to verify that $(\mathcal{C}^x)^y = (\mathcal{C}^y)^x$; therefore, for a set of variables $V = \{x_1, \ldots, x_r\}$ we can unambiguously define $\mathcal{C}^V \stackrel{\text{def}}{=} ((\mathcal{C}^{x_1})^{x_2} \ldots)^{x_r}$.

Intuitively, $\mathcal{C}^V$ is all "information" expressible in the variables $Vars(\mathcal{C}) \setminus V$ that we can extract from $\mathcal{C}$.

For simplicity we will identify the clauses in τ with the corresponding hyperedges in $\mathcal{H}_\tau$. Our algorithm traverses the nodes of T in an upward (from leaves to the root) order, say, by the depth-first search. For every visited node t the algorithm recursively constructs a set of clauses $\mathcal{C}(t)$ in the variables $\mathrm{Cut}(t)$ as shown on Figure 2.

Since $|\mathrm{Cut}(t)| \leq w(T, \theta)$ for every node, $|\mathcal{C}(t)| \leq 2^{O(w(T,\theta))}$; thus, this algorithm works in time $|\tau|^{O(1)} \cdot 2^{O(w(T,\theta))}$ and produces a resolution inference of $\mathcal{C}(root)$ (the latter is either an empty set or contains just the empty clause) of width $O(w(T, \theta))$. Next, it is easy to see that if s is not a descendant of t then

$$V'(t) \cap V(s) = \emptyset. \tag{3}$$

In particular, $V'(t) \cap V'(s) = \emptyset$ for every two different nodes t, s which implies that this inference is regular.

Finally we show that if τ is unsatisfiable then $\emptyset \in \mathcal{C}(root)$. For this purpose, we will keep track of the following auxiliary CNF τ_{current} during the progress of the algorithm. Initially $\tau_{\text{current}} = \cup \{\mathcal{C}(t) \mid t \text{ is a leaf}\}$, and every

```
  begin
    for all leaves t in T do
    if θ^{-1}(t) contains a variable that does not appear in any other clause
    then C(t) := ∅ else C(t) := {θ^{-1}(t)};
    for all internal nodes t in T visited in an upward order do
    begin
        t_1, t_2 := the two children of t;
        V'(t) := (Cut(t_1) ∪ Cut(t_2)) \ Cut(t);
(1)     C(t) := (C(t_1) ∪ C(t_2))^{V'(t)};
    end
end
```

Figure 2: Converting branch-decompositions into regular resolution refutations

time the operator **(1)** is applied, we replace the clauses $\mathcal{C}(t_1) \cup \mathcal{C}(t_2)$ in τ_{current} with $\mathcal{C}(t)$. We claim that τ_{current} stays unsatisfiable all the time.

Indeed, the initial value $\cup \{\mathcal{C}(t) \mid t \text{ is a leaf}\}$ of τ_{current} is obtained from τ by removing all clauses that contain a variable not appearing elsewhere. Clearly, unsatisfiability of this shorter CNF is implied by unsatisfiability of τ (every removed clause can be satisfied by appropriately setting its individual variable).

During every individual step **(1)** we replace τ_{current} by $\tau'_{\text{current}} \stackrel{\text{def}}{=} (\tau_{\text{current}} \setminus (\mathcal{C}(t_1) \cup \mathcal{C}(t_2))) \cup (\mathcal{C}(t_1) \cup \mathcal{C}(t_2))^{V'(t)}$. The observation (3), however, implies that $Vars(\tau_{\text{current}} \setminus (\mathcal{C}(t_1) \cup \mathcal{C}(t_2))) \cap V'(t) = \emptyset$, therefore $\tau'_{\text{current}} = (\tau_{\text{current}})^{V'(t)}$. Now the inductive step immediately follows from the following simple fact (see e.g. [Kra95, Theorem 4.2.1]):

Statement 3.4 *If τ is an unsatisfiable CNF, and x is a variable then τ^x is unsatisfiable.*

At the end of our algorithm, $\tau_{\text{current}} = \mathcal{C}(root)$, and its unsatisfiability implies $\emptyset \in \mathcal{C}(root)$.

This completes the proof of Lemma 3.2. It is worth noting that, although it is not quite immediate from the description, the proof reveals that refutation we construct actually belongs to the class of Davis-Putnam procedures, as defined e.g. in [BG99].■

As we already observed above, Theorem 2.2 immediately follows from Lemmas 3.1 and 3.2, it is left to notice that if the algorithm gets stuck at some point (e.g. $\mathcal{C}(root)$ does not contain any clauses) then we can trace the process back and inductively find a satisfying assignment.

4. Width automatizability of Tseitin tautologies

Now we prove Theorem 2.7. For this we need the following notion of Gaussian width (although it makes perfect sense for arbitrary systems of linear equations, we formulate it here only for the partial case of Tseitin tautologies).

Definition 4.1 For an ordinary graph G and $V \subseteq V(G)$, let $\mathrm{Cut}_G(V) \stackrel{\mathrm{def}}{=} \{(v, v') \in E(G) \mid v \in V,\ v' \notin V\}$.

Definition 4.2 A line in the *Gaussian calculus associated with the Tseitin tautology* $T(G, \sigma)$ is a linear equation mod 2 of the form

$$L_V \stackrel{\mathrm{def}}{=} \left(\bigoplus_{e \in \mathrm{Cut}_G(V)} x_e = \bigoplus_{v \in V} \sigma(v) \right),$$

where $V \subseteq V(G)$. L_V is the sum mod 2 of the initial axioms (1); note that since G is connected and σ is odd-weight, V is uniquely determined by L_V. The only inference rule of the Gaussian calculus is

$$\frac{L_{V_1} \qquad L_{V_2}}{L_{V_1 \triangle V_2}}.$$

A *Gaussian refutation of* $T(G, \sigma)$ is a proof of $L_{V(G)}$ (i.e., $0 = 1$) from the axioms L_v (i.e., (1)) in the Gaussian calculus associated with $T(G, \sigma)$. The *width of the line* L_v is the number $|\mathrm{Cut}_G(V)|$ of its variables, the *width of a Gaussian refutation* is the maximum width of all its lines, and the *Gaussian width of* $T(G, \sigma)$, denoted by $w_g(G, \sigma)$ is the minimum width of a Gaussian refutation of $T(G, \sigma)$.

The following was observed by the first author during the work on the paper [ABRW00] and successfully used in [BI99] for characterizing the degree of polynomial calculus refutations of systems of linear equations mod 2.

Proposition 4.3 $\frac{1}{2}w(T(G,\sigma)\vdash\emptyset) \leq w_g(G,\sigma) \leq w(T(G,\sigma)\vdash\emptyset)$.

Now we are ready to prove Theorem 2.7. Given Proposition 4.3, we have to show that $w_b(T(G,\sigma)) = \theta(w_g(G,\sigma))$. As we already observed, the hypergraph $\mathcal{H}_{T(G,\sigma)}$ associated with the CNF $T(G,\sigma)$ is just G^*, with the difference that every hyperedge repeats itself several times. It is easy to see that removing repeated occurrences of edges from a hypergraph does not change its branch-width. Thus, the theorem now follows from the following statement:

Lemma 4.4 $w_b(G^*) = \theta(w_g(G,\sigma))$.

Proof. By examining definitions, one can see that $w_b(G^*)$ is actually the same as $w_g(G,\sigma)$, with the only difference that we are allowed only *read-once* (tree-like) Gaussian refutations, i.e., those refutations in which every axiom (1) is used exactly once. This remark proves $w_b(G^*) \geq w_g(G,\sigma)$.

For the opposite direction, we once more refer to the analysis of the algorithm shown on Figure 1. First note that (2) can be obviously generalized to $w_g(G,\sigma) \geq r(G^*)$ ($r(G^*)$ is the maximum vertex degree in G). Therefore, like in the proof of Lemma 3.1, it suffices to show that the obstacle[3] $Z \subseteq E(G)$ making our algorithm to abort at the $(w-1)$th stage implies the stronger inequality $w \leq O(w_g(G,\sigma))$. This is done by the straightforward replacement of the optimal branch-decomposition (T,θ) with the (also optimal) Gaussian refutation P of $T(G,\sigma)$ that has Gaussian width $w_g(G,\sigma)$. Let T be the underlying tree of P, and, for a node t of T let $L_{V(t)}$ be the line sitting in that node. Next, let $E(t) \stackrel{\text{def}}{=} \bigcup_{v\in V(t)} S_G(v)$ and, similarly, $E^\perp(t) \stackrel{\text{def}}{=} \bigcup_{v\in co-V(t)} S_G(v)$ (note that $\mathrm{Cut}_G(V(t)) = E(t)\cap E^\perp(t)$.) Like before, denote $Z(t) \stackrel{\text{def}}{=} Z\cap E(t)$ and $Z^\perp(t) \stackrel{\text{def}}{=} Z\cap E^\perp(t)$. If t is an internal node of T and t_1, t_2 its children, then $V(t) = V(t_1)\triangle V(t_2) \subseteq V(t_1)\cup V(t_2)$, hence we still have $E(t) \subseteq E(t_1)\cup E(t_2)$ and $|Z(t)| \leq |Z(t_1)| + |Z(t_2)|$. Arguing in the same way as before, we find a node t such that $|Z(t)| \geq \frac{1}{3}|Z|$ and $|Z^\perp(t)| \geq \frac{1}{3}|Z|$. The set of edges appearing in $L_{V(t)}$ is $\mathrm{Cut}_G(V(t))$, and this

[3] it is important to bear in mind that the notation in the proof of that lemma corresponds to the dual hypergraph G^*, and now vertices and edges have exchanged places

is an edge cut separating $Z(t) \setminus \mathrm{Cut}_G(V(t))$ from $Z^{\perp}(t) \setminus \mathrm{Cut}_G(V(t))$. The rest of the proof repeats the previous argument.$\blacksquare$

5. (Quasi-)smooth classes of Tseitin tautologies

In this section we prove Theorems 2.10, 2.12 and 2.13. All proofs follow the same ideology proposed in [BP96]. Namely, we are going to define a random restriction $\boldsymbol{\rho}$ that will have the following two properties:

1. with high probability, $\boldsymbol{\rho}$ does not reduce $w(T(G,\sigma) \vdash \emptyset)$ too much;

2. every clause of large width is killed (also with high probability).

Then property 1 will allow us to upper bound $w(T(G,\sigma) \vdash \emptyset)$ in terms of $w(T(G,\sigma)|_{\boldsymbol{\rho}} \vdash \emptyset)$, and property 2 will give an upper bound on the latter quantity in terms of $S(\tau)$.

Proof of Theorem 2.10. The proof uses a construction similar to that in [Ale01].

Let us fix a family $\mathcal{C}$ of cycles in G that have length $\leq \ell(G)$ and such that every edge belongs to at least 1 and at most $\ell(G)$ cycles in the family. Let us make the auxiliary graph $L_{\mathcal{C}}(G)$ whose vertices are edges of G, e and e' being connected in $L_{\mathcal{C}}(G)$ if and only if e and e' appear together in at least one cycle $C \in \mathcal{C}$. Note that

$$\deg_{L_{\mathcal{C}}(G)}(e) \leq \ell(G) \cdot (\ell(G) - 1), \tag{4}$$

for all $e \in E(G)$.

Claim 5.1 *Let ρ be any restriction that arbitrarily assigns to 0,1 the variables $\{x_e \mid e \in E\}$, where E is an independent set in $L_{\mathcal{C}}(G)$ (i.e., no two edges appear in the same cycle $C \in \mathcal{C}$). Then for any odd-weight σ, $w(T(G,\sigma) \vdash \emptyset) \leq O(\ell(G) \cdot w(T(G,\sigma)|_{\rho} \vdash \emptyset))$.*

Proof of Claim 5.1. First of all note that $T(G,\sigma)|_{\rho} = T(\tilde{G},\tilde{\sigma})$, where $\tilde{G} = (V(G), E(G) \setminus E)$, and $\tilde{\sigma}$ is a naturally defined adjustment of σ by the values assigned by ρ. Because of Theorem 2.7 (and a remark at the beginning of its proof), it suffices to show that $w_b(G^*) \leq O(\ell(G) \cdot w_b(\tilde{G}^*))$. Let us fix a branch-decomposition (T,θ) of $\tilde{G}^*$ that has width $w_b(\tilde{G}^*)$. Since G and $\tilde{G}$ have the same set of vertices, (T,θ) can also be considered as a branch-decomposition of G^*, and we claim that it has width $O(\ell(G)\cdot w_b(\tilde{G}^*))$.

Indeed, for any node t of T, $|\mathrm{Cut}_{G^*}(t)| = |\mathrm{Cut}_{\tilde{G}^*}(t)| + |\mathrm{Cut}_{G^*}(t)\cap E|$. For every $e \in \mathrm{Cut}_{G^*}(t)\cap E$, pick up an arbitrary cycle $C_e \in \mathcal{C}$ containing e. Note that since E is independent in $L_{\mathcal{C}}(G)$, all these cycles are pairwise different. On the other hand, C_e must contain another edge $f_e \in \mathrm{Cut}_{G^*}(t)$ which is not in E. Every edge $f \in E(G)\setminus E$ may appear as f_e in at most $\ell(G)$ cycles C_e, therefore $\frac{|\mathrm{Cut}_{G^*}(t)\cap E|}{\ell(G)} \leq |\mathrm{Cut}_{\tilde{G}^*}(t)|$ and then $|\mathrm{Cut}_{G^*}(t)| \leq (\ell(G)+1)\cdot|\mathrm{Cut}_{\tilde{G}^*}(t)| \leq (\ell(G)+1)\cdot w_b(\tilde{G}^*)$. Claim 5.1 is proved.■

Let us call $E \subseteq E(G)$ *strongly independent* if the distance between every two distinct $e, e' \in E$ in the graph $L_{\mathcal{C}}(G)$ is at least three.

Claim 5.2 *Every $E \subseteq E(G)$ contains a strongly independent subset $E_0 \subseteq E$ with $|E_0| \geq |E|/\ell^4(G)$.*

Proof of Claim 5.2. Construct E_0 by a greedy algorithm using the vertex degree bound (4).■

Now we are ready to finish the proof of Theorem 2.10. Let $\ell \stackrel{\mathrm{def}}{=} \ell(G)$, $p \stackrel{\mathrm{def}}{=} 1/(2\ell^2)$, and let $\boldsymbol{E_p} \subseteq E(G)$ be a random subset of edges in which every edge appears, independently of others, with probability p. Let $\boldsymbol{E} \stackrel{\mathrm{def}}{=} \left\{ e \in \boldsymbol{E_p} \,\middle|\, e \text{ is isolated in } L_{\mathcal{C}}(G)|_{\boldsymbol{E_p}} \right\}$. Note that $\boldsymbol{E}$ is always an independent set in $L_{\mathcal{C}}(G)$. Let $\boldsymbol{\rho}$ be a random restriction that assigns at random the variables $\{x_e \mid e \in \boldsymbol{E}\}$. Then Claim 5.1 implies that (with probability 1)

$$w(T(G,\sigma)\vdash\emptyset) \leq O(\ell\cdot w(T(G,\sigma)|_{\boldsymbol{\rho}}\vdash\emptyset)). \tag{5}$$

On the other hand, let P be a resolution refutation of $T(G,\sigma)$ that has size S, and let $C \in P$ be any clause in it of width $\geq K\ell^6\log S$, $K>0$ is a sufficiently large constant. Applying Claim 5.2, find $D \leq C$ of width $K\ell^2\log S$ such that $E_0 \stackrel{\mathrm{def}}{=} \{e \in E(G) \mid x_e \in Vars(D)\}$ is strongly independent. The event $e \in \boldsymbol{E}$ depends only on the behavior of $\boldsymbol{E_p}$ on the star $S_{L_{\mathcal{C}}(G)}(e)$ of the edge e in the graph $L_{\mathcal{C}}(G)$, and these stars are disjoint for different $e \in E_0$ (since E_0 is strongly independent.) Thus, the events $\{e \in \boldsymbol{E} \mid e \in E_0\}$ are independent, and each of them has probability $\mathbf{P}[e \in \boldsymbol{E}] = \mathbf{P}[e \in \boldsymbol{E_p}]\,\cdot$

$\mathbf{P}\Big[S_{L_{\mathcal{C}}(G)}(e) \cap \boldsymbol{E_p} = \emptyset\Big] \geq p \cdot (1 - p \cdot \deg_{L_{\mathcal{C}}(G)}(e)) \geq \Omega(\ell^{-2})$. Hence, by Chernoff's inequality,

$$\mathbf{P}[|\boldsymbol{E} \cap E_0| \leq 2\log_2 S] \leq S^{-2}, \tag{6}$$

provided the constant K is large enough. Therefore, $\mathbf{P}[C|_{\boldsymbol{\rho}} \not\equiv 1] \leq \mathbf{P}[D|_{\boldsymbol{\rho}} \not\equiv 1] \leq \mathbf{P}[D|_{\boldsymbol{\rho}} \not\equiv 1 \,|\, |\boldsymbol{E} \cap E_0| \geq 2\log_2 S] + S^{-2} \leq 2S^{-2}$. This implies that with the overwhelming probability $\boldsymbol{\rho}$ kills all clauses in P of width $\geq K\ell^6 \log S$, and this, in turn, implies that (also with the overwhelming probability)

$$w(T(G,\sigma)|_{\boldsymbol{\rho}} \vdash \emptyset)) \leq O(\ell^6 \log S). \tag{7}$$

Theorem 2.10 now immediately follows from (5) and (7).■

Proof of Theorem 2.12. The main complication we are facing now is that we do not have any bounds on the vertex degrees in the graph $L_{\mathcal{C}}(G)$. Respectively, we can not successfully employ (strongly) independent sets in this graph as in the proof of Theorem 2.10. We are going to replace this by a much more elaborate combinatorial analysis based upon the sunflower theorem.

Definition 5.3 A *sunflower of size* r is any family of sets $\{F_1, \ldots, F_r\}$ (called *petals*) such that for all pairs $1 \leq i < j \leq r$, the intersection $F_i \cap F_j$ has one and the same value which we will call the *kernel* of the sunflower and denote by $\mathrm{Ker}(F_1, \ldots, F_r)$.

Proposition 5.4 ([ER60]) *Every family of sets $\mathcal{F}$ such that $\forall F \in \mathcal{F}(|F| \leq \ell)$ and $|\mathcal{F}| > \ell!(r-1)^\ell$ contains a sunflower of size r.*

We begin the formal proof of Theorem 2.12 with the following weak form of Claim 5.2.

Claim 5.5 *Every $E \subseteq E(G)$ such that*

$$|E| \geq (2\tilde{\ell}^2(G))^{\tilde{\ell}(G)} \tag{8}$$

contains $E_0 \subseteq E$ such that the edges of E_0 can be covered by edge-disjoint cycles of length $\leq 2\tilde{\ell}(G)$ and $|E_0| \geq \frac{1}{4\tilde{\ell}(G)}|E|^{1/\tilde{\ell}(G)}$.

Proof of Claim 5.5. For every $e \in E$, fix a cycle C_e of length at most $\ell \stackrel{\mathrm{def}}{=} \tilde{\ell}(G)$ such that $e \in C_e$. By Proposition 5.4, $\{C_e \mid e \in E\}$ must contain a sunflower [4] $\{C_e \mid e \in E'\}$ of size $\frac{1}{\ell}|E|^{1/\ell}$. Note that (8) implies $|E'| \geq 2\ell$. Let

[4]note that for the sake of this argument we completely disregard the structure existing on $E(G)$ and treat cycles simply as sets of edges

$E'' \stackrel{\rm def}{=} E' \setminus \mathrm{Ker}(\{ C_e \mid e \in E' \})$; then $|E''| \geq \frac{1}{2\ell}|E|^{1/\ell}$. Enumerate the elements of E'' in an arbitrary order $E'' = \{e_1, e_2, \ldots, e_{2w}\}$, where $w \stackrel{\rm def}{=} \frac{1}{4\ell}|E|^{1/\ell}$ and form the symmetric differences $C_{e_1} \triangle C_{e_2},\ C_{e_3} \triangle C_{e_4}, \ldots, C_{e_{2w-1}} \triangle C_{e_{2w}}$. They are edge-disjoint (since $\{ C_e \mid e \in E' \}$ is a sunflower), and $C_{e_{2\nu-1}} \triangle C_{e_{2\nu}}$ contains a simple cycle going through $e_{2\nu-1}$ (since $e_{2\nu-1} \not\in \mathrm{Ker}(\{ C_e \mid e \in E' \})$). Thus, $E_0 \stackrel{\rm def}{=} \{e_1, e_3, \ldots, e_{2w-1}\}$ has all the required properties. Claim 5.5 is completely proved.■

Our next step is to make as minimal adjustment of $\boldsymbol{E_p}$ as possible. Namely, we will keep in it an edge e as long as *there exists at least one* short cycle C such that $C \cap \boldsymbol{E_p} = \{e\}$ (in the previous proof we demanded that *all cycles in* $\mathcal{C}$ containing e have this property). This will allow us to estimate the probability of killing a large clause quite easily, but will create new complications in proving the analogue of Claim 5.1.

Definition 5.6 For $E \subseteq E(G)$ and $\ell > 0$, let $\partial_\ell(E)$ consist of those $e \in E$ for which there exists a cycle of length $\leq \ell$ with $C \cap E = \{e\}$ (equivalently, the distance between the endpoints of e in the graph $(V(G), E(G) \setminus E)$ is at most $\ell - 1$).

Let $\ell \stackrel{\rm def}{=} \tilde{\ell}(G)$, $p = 1/(4\ell)$ and $\boldsymbol{E} \stackrel{\rm def}{=} \partial_{2\ell}(\boldsymbol{E_p})$. Let C be any clause of width $\geq (K\ell^2 \log S)^\ell$, where K is a sufficiently large constant. Pick $D \leq C$ of width $\geq (K/4)\ell \log S$ according to Claim 5.5, let $E_0 \stackrel{\rm def}{=} \{ e \mid x_e \in Vars(D) \}$ and $\{ C_e \mid e \in C_0 \}$ be the family of edge-disjoint cycles of length $\leq 2\ell$ such that $e \in C_e$. Then $C_e \cap \boldsymbol{E_p} = \{e\}$ implies $e \in \boldsymbol{E}$, the events $\{ C \cap \boldsymbol{E_p} = \{e\} \mid e \in E_0 \}$ are independent, and each has the probability $\geq 1/(8\ell)$. As before, by Chernoff's bound we have (6) which allows us to conclude

$$\mathbf{P}\Big[w(T(G,\sigma)|_{\boldsymbol{\rho}} \vdash \emptyset)) \geq (K\ell^2 \log S)^\ell \Big] \leq O(S^{-1}), \tag{9}$$

where, as before, $\boldsymbol{\rho}$ is a random restriction which at random assigns to $\{0,1\}$ the variables $\{ x_e \mid e \in \boldsymbol{E} \}$.

Let us now turn to the analogue of Claim 5.1. Arguing as in the proof of that claim, we only have to rule out the bad event consisting in the existence of a set of vertices $V \subseteq V(G)$ such that $\mathrm{Cut}_G(V)$ is large whereas $\mathrm{Cut}_{\tilde{\boldsymbol{G}}}(V)$ is small, where, as before, $\tilde{\boldsymbol{G}} \stackrel{\rm def}{=} (V(G), E(G) \setminus \boldsymbol{E})$. We will do this by showing that its probability is negligible.

begin
$\mathcal{F} := \mathcal{P}_{V_0,V_1}$;
while $\mathcal{F}$ contains r *minimal* (w.r.t. inclusion) elements $F_1, \ldots, F_r$ that form a sunflower **do** $\mathcal{F} := \mathcal{F} \setminus \{F_1, \ldots, F_r\} \cup \{\mathrm{Ker}(F_1, \ldots, F_r)\}$;
$\mathcal{F}_{V_0,V_1} :=$ the set of minimal elements in $\mathcal{F}$;
return $\mathcal{F}_{V_0,V_1}$;
end

Figure 3: Plucking procedure

Assume for a moment that this bad event has happened. Pick an arbitrary $\boldsymbol{V}$ such that $\mathrm{Cut}_G(\boldsymbol{V})$ is large and $\mathrm{Cut}_{\tilde{\boldsymbol{G}}}(\boldsymbol{V})$ is small. Let $\boldsymbol{V_0}$ be the set of those endpoints of edges in $\mathrm{Cut}_{\tilde{\boldsymbol{G}}}(\boldsymbol{V})$ that belong to $\boldsymbol{V}$. Likewise, let $\boldsymbol{V}_1$ be their endpoints in $V(G) \setminus \boldsymbol{V}$. Note that $|\boldsymbol{V_0}|, |\boldsymbol{V_1}| \leq |\mathrm{Cut}_{\tilde{\boldsymbol{G}}}(\boldsymbol{V})|$ are small. We are going to show that for every *fixed* choice of $\boldsymbol{V_0}, \boldsymbol{V_1}$ the above situation may happen only with a very small probability (and then we apply the union bound). For doing this we will note that the sets $\boldsymbol{V}_0$ and $\boldsymbol{V}_1$ represent endpoints of a cut in $\tilde{\boldsymbol{G}}$. As we will show, every edge in $Cut_G(V) \setminus Cut_{\tilde{\boldsymbol{G}}}$ belongs to some path that connects points $v_0 \in \boldsymbol{V}_0$ and $v_1 \in \boldsymbol{V}_1$. But the number of such paths can be very large and the structure complicated, thus we will use sunflowers once more to simplify the picture.

Returning to the formal argument, let us fix disjoint $V_0, V_1 \subseteq V(G)$. Let $\mathcal{P}_{V_0,V_1}$ be the set of all simple paths P connecting a vertex in V_0 to a vertex in V_1, having length at least 2 and at most $(2\ell-1)$ and such that no internal node of P is in $V_0 \cup V_1$. Let r be a parameter to be fixed later. We apply to $\mathcal{P}_{V_0,V_1}$ the *plucking procedure* shown on Figure 3 in the same way it was used in [Raz85, AB87] for lower bounds on the monotone circuit size.

Let us call $e \in E(G)$ *relevant* (to (V_0, V_1)) if it appears in at least one $F \in \mathcal{F}_{V_0,V_1}$. Every $F \in \mathcal{F}_{V_0,V_1}$ can be inferred from $\mathcal{P}_{V_0,V_1}$ using the *sunflower inference rule*

$$\frac{F_1, \ldots, F_r}{\mathrm{Ker}(F_1, \ldots, F_r)} \ ((F_1, \ldots, F_r) \text{ a sunflower}). \tag{10}$$

Let us call an instance of the rule (10) *relevant* if it is used in the inference of at least one $F \in \mathcal{F}_{V_0,V_1}$. Finally, let us call a set of edges E *regular* (again, with respect to (V_0, V_1)) if for every **relevant** instance (10) there exists $\nu \in [r]$

such that $E \cap (F_\nu \setminus \mathrm{Ker}(F_1, \ldots, F_r)) = \emptyset$. Then the combinatorial essence of our argument can be extracted as follows.

Claim 5.7 *Let V_0, V_1 be disjoint subsets of $V(G)$, $\hat{E}$ be any set of edges that is regular w.r.t. (V_0, V_1), $E \stackrel{\mathrm{def}}{=} \partial_{2\ell}(\hat{E})$ and $\tilde{G} \stackrel{\mathrm{def}}{=} (V(G), E(G) \setminus E)$. Then for every $V \subseteq V(G)$ such that $V_0 \subseteq V$, $V_1 \cap V = \emptyset$ and $\mathrm{Cut}_{\tilde{G}}(V) \subseteq V_0 \times V_1$, all edges in $\mathrm{Cut}_G(V) \setminus (V_0 \times V_1)$ are relevant to (V_0, V_1).*

Proof of Claim 5.7. Fix any V with the properties stated above, and let $e \in \mathrm{Cut}_G(V) \setminus (V_0 \times V_1)$. Since $e \in E$, there exists a cycle C_e of length $\leq 2\ell$ such that $C_e \cap E = \{e\}$. Let P_e be the maximal subpath of C_e such that no internal vertex of P_e belongs to $V_0 \cup V_1$ and $P_e \cap \mathrm{Cut}_G(V) = \{e\}$. We claim that both end-points of P_e are in $V_0 \cup V_1$. Indeed, let v be one of them, and f is the edge in $C_e \setminus P_e$ adjacent to v. Due to the maximality of P_e, f must violate one of the conditions above. If adding f creates an internal point in $V_0 \cup V_1$, we are done. Assume $f \in \mathrm{Cut}_G(V)$. Note that $f \notin E$ (since e is the only edge in the intersection $C_e \cap E$). Therefore, $f \in V_0 \times V_1$ which implies that in this case we also have $v \in V_0 \cup V_1$. We have proved that both end-points of P_e belong to $V_0 \cup V_1$, and this in particular implies $P_e \in \mathcal{P}_{V_0,V_1}$.

Next, there exists $F_e \subseteq P_e$ such that $F_e \in \mathcal{F}_{V_0,V_1}$, and we only have to prove that in fact $e \in F_e$. And for that it is sufficient to show that $F_e \cap E \neq \emptyset$ (since $F_e \subseteq P_e \subseteq C_e$ may contain only one edge in E which is e).

This, however, is easy. Namely, since $\hat{E}$ is regular, E is regular, too, which implies (by an easy induction on the inference of F_e from $\mathcal{P}_{V_0,V_1}$ using relevant inference rules) that there exists $P'_e \in \mathcal{P}_{V_0,V_1}$ such that $P'_e \supseteq F_e$ and $P'_e \cap E = F_e \cap E$. But every element of $\mathcal{P}_{V_0,V_1}$ must contain at least one edge which is in $\mathrm{Cut}_G(V)$ but not in $V_0 \times V_1$ (remember that we required the length of every path in $\mathcal{P}_{V_0,V_1}$ to be at least 2), and this edge must be in E. This shows that indeed $F_e \cap E \neq \emptyset$, that e is relevant and completes the proof of Claim 5.7.■

Now we can easily complete the proof of Theorem 2.12. Let

$$w \stackrel{\mathrm{def}}{=} (K\ell^2 \log S)^\ell$$

be the upper bound on $w(T(G, \sigma)|_{\boldsymbol{\rho}} \vdash \emptyset)$ in (9). Set

$$r \stackrel{\mathrm{def}}{=} Cw \log n, \tag{11}$$

where $n \stackrel{\rm def}{=} |V(G)|$ and $C > 0$ is a sufficiently large constant. By Proposition 5.4, $|\mathcal{F}_{V_0,V_1}| \leq (\ell r)^{O(\ell)} \leq (\ell \log S)^{O(\ell^2)}$, where for the last inequality we used the trivial bound $n \leq S$. Therefore, for every particular pair (V_0, V_1) the number of relevant edges is also bounded by $(\ell \log S)^{O(\ell^2)}$.

Let us call E *regular* if it is regular w.r.t. any pair (V_0, V_1) of disjoint subsets of $V(G)$ such that $|V_0|, |V_1| \leq w$. Then Claim 5.7 (applied to $\hat{E} := \boldsymbol{E_p}$) implies that if $\boldsymbol{E_p}$ is regular then for every V, $|\mathrm{Cut}_{\tilde{\boldsymbol{G}}}(V)| \leq w \Longrightarrow |\mathrm{Cut}_G(V)| \leq w^2 + (\ell \log S)^{O(\ell^2)} \leq (\ell \log S)^{O(\ell^2)}$. As we remarked above, this implies the bound

$$w_b(T(G,\sigma)|_{\boldsymbol{\rho}} \vdash \emptyset) \leq w \Longrightarrow w_b(T(G,\sigma) \vdash \emptyset) \leq (\ell \log S)^{O(\ell^2)}, \qquad (12)$$

as long as $\boldsymbol{E_p}$ is regular. We are only left to show that this happens with the overwhelming probability.

First, for any particular instance of the rule (10),

$$\begin{aligned} &\mathbf{P}[\forall \nu \in \{1, \ldots, r\} \boldsymbol{E_p} \cap (F_\nu \setminus \mathrm{Ker}(F_1, \ldots, F_r)) \neq \emptyset] \\ &\leq \prod_{\nu=1}^{r} (p|F_\nu \setminus \mathrm{Ker}(F_1, \ldots, F_r)|) \leq 2^{-r}. \end{aligned}$$

Next, the inference of any $F \in \mathcal{F}_{V_0,V_1}$ from $\mathcal{P}_{V_0,V_1}$ can be represented as a tree of height $\leq 2\ell$ and degree r. Therefore, it uses at most $r^{O(\ell)}$ applications of the inference rule. Thus, for a given pair (V_0, V_1), the overall number of relevant instances is $(\ell w \log n)^{O(\ell)} \leq (\ell \log S)^{O(\ell^2)}$. Finally, there are at most $n^{O(w)}$ choices of the pair (V_0, V_1), and

the overall number of instances relevant to at least one of them is $(\ell \log S)^{O(\ell^2)} \cdot n^{O(w)} \leq n^{O(w)}$. Summing up, we get $\mathbf{P}[\boldsymbol{E_p}$ is not regular$] \leq 2^{-r} n^{O(w)} \leq o(1)$ if the constant C in (11) is large enough.

We have proved that the event (12) happens with probability $\geq 1 - o(1)$. Comparing this with the previous conclusion (9) completes the proof of Theorem 2.12.■

Proof of Theorem 2.13. This time, let $\boldsymbol{\rho}$ simply assign at random the variables $\left\{ x_e \,\middle|\, e \in \boldsymbol{E_{1/2}} \right\}$. Then $\mathbf{P}[w_b(T(G,\sigma)|_{\boldsymbol{\rho}} \vdash \emptyset) \leq C \cdot \log_2 S] \geq 1 - o(1)$, C a sufficiently large constant. On the other hand, for every V with $|\mathrm{Cut}_G(V)| \geq C^2(\log_2 S + n)$ we can apply Chernoff's bound to conclude that $\mathbf{P}[|\mathrm{Cut}_{\tilde{\boldsymbol{G}}}(V)| < C \log_2 S] \leq 2^{-2n}$. Since there are 2^n choices of V, we conclude by the union bound that

$$\mathbf{P}\Big[w_b(T(G,\sigma)|_{\boldsymbol{\rho}} \vdash \emptyset) \leq C \cdot \log_2 S \Longrightarrow w_b(T(G,\sigma) \vdash \emptyset) \leq C^2(\log_2 S + n)\Big]$$
$$\geq 1 - 2^{-n}.$$

Theorem 2.13 follows.■

6. Open problems

The first important problem is to overcome the main difficulty of the practical implementation which is the huge amount of space used by width-based algorithms: the space required by both WBATP and BWBATP is roughly the same as their running time. It is not clear how to store in the memory (a priori arbitrary) Boolean functions over $w_b(\tau)$ variables that appear at each step of our algorithm. Thus we ask if one can do anything intelligent in polynomial space to check satisfiability of formulas with small branch-width?

Another interesting question is to try to further narrow the gap between positive and negative results. Except for obvious motivations (constructing new automated theorem provers), that might also shed some light on the important problem left open in [AR01b]: prove that Resolution is not quasi-automatizable (or that it is not automatizable under a complexity assumption substantially weaker than the one used in [AR01b]).

The definition of Tseitin tautologies $T(G,\sigma)$ can be naturally generalized in two possible ways: G can be replaced by a hypergraph $\mathcal{H}$ and counting mod 2 can be replaced by counting mod p or over integers (see e.g. [ABRW00, BI99, AR01a] for details). Is either of the two resulting classes width automatizable (for the first generalization it also makes sense to ask the same question when $r(\mathcal{H})$ is bounded)?

Finally, is the class of all (ordinary) Tseitin tautologies smooth or quasi-smooth? This question is closely related to the hypothesis of Alasdair Urquhart, who conjectured that regular Resolution can efficiently simulate general Resolution on Tseitin tautologies.

7. Acknowledgements

We are grateful to both anonymous referees of our FOCS02 submission for their useful remarks.

References

[Ale01] M. Alekhnovich. Mutilated chessboard problem is exponentially hard for Resolution. Manuscript.

[ABRW00] M. Alekhnovich, E. Ben-Sasson, A. Razborov, and A. Wigderson. Pseudorandom generators in propositional complexity. In *Proceedings of the* 41*st IEEE FOCS*, pages 43–53, 2000.

[AR01a] M. Alekhnovich and A. Razborov. Lower bounds for the polynomial calculus: non-binomial case. In *Proceedings of the* 42*nd IEEE Symposium on Foundations of Computer Science*, pages 190–199, 2001.

[AR01b] M. Alekhnovich and A. Razborov. Resolution is not automatizable unless $W[P]$ is tractable. In *Proceedings of the* 42*nd IEEE Symposium on Foundations of Computer Science*, pages 210–219, 2001.

[AB87] N. Alon and R. Boppana. The monotone circuit complexity of Boolean functions. *Combinatorica*, 7(1):1–22, 1987.

[AM01] E. Amir and S. McIlraith. Solving satisfiability using decomposition and the most constrained subproblem. *LICS workshop on Theory and Applications of Satisfiability Testing (SAT 2001)*, 2001.

[AB02] A. Atserias and M. Bonet. On the automatizability of resolution and related propositional proof systems. Technical Report TR02-010, Electronic Colloquium on Computational Complexity, 2002.

[BP96] P. Beame and T. Pitassi. Simplified and improved resolution lower bounds. In *Proceedings of the* 37*th IEEE FOCS*, pages 274–282, 1996.

[BI99] E. Ben-Sasson and R. Impagliazzo. Random CNF"s are hard for the polynomial calculus. In *Proceedings of the* 40*th IEEE FOCS*, pages 415–421, 1999.

[BW01] E. Ben-Sasson and A. Wigderson. Short proofs are narrow - resolution made simple. *Journal of the ACM*, 48(2):149–169, 2001.

[Bod93] H. L. Bodlaender. A tourist guide through treewidth. *Acta Cybernetica*, 11: 1–21, 1993.

[BG99] M. Bonet and N. Galesi. A study of proof search algorithms for Resolution and Polynomial Calculus. In *Proceedings of the 40th IEEE FOCS*, pages 422–431, 1999.

[BPR00] M. Bonet, T. Pitassi, and R. Raz. On interpolation and automatization for Frege systems. *SIAM Journal on Computing*, 29(6):1939–1967, 2000.

[CMR01] B. Courcelle, J.A. Makowsky and U. Rotics. On the fixed parameter complexity of graph enumeration problems definable in monadic second order logic. *Discrete Applied Mathematics,* 108 (1-2) :23–52, 2001.

[ER60] P. Erdös and R. Rado. Intersection theorems for systems of sets. *Journal of the London Math. Society*, 35:85–90, 1960.

[Kra95] J. Krajíček. *Bounded arithmetic, propositional logic and complexity theory.* Cambridge University Press, 1995.

[KM96] S. Khanna and R. Motwani. Towards a syntactic characterization of PTAS. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing*, pages 329–337, 1996.

[LT79] R. Lipton and R. Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36: 177-189, 1979.

[LT80] R. Lipton and R. Tarjan. Applications of a planar separator theorem. *SIAM Journal on Computing*, 9:615-627, 1980.

[Raz85] A. A. Razborov. Lower bounds for the monotone complexity of some boolean functions. *Doklady Academii Nauk SSSR*, 281(4):798–801, 1985. English Translation in *Soviet Math. Dokl.*, 31:354-357, 1985.

[RS91] N. Robertson and P. D. Seymour. Graph Minors X. Obstructions to tree decomposition. *Journal of Combinatorial Theory Series B*, 52:153–190, 1991.

[RS95] N. Robertson and P. D. Seymour. Graph minors. XIII. The disjoint paths problem. *Journal of Combinatorial Theory Series B*, 63:65–110, 1995.

[Tse68] G. C. Tseitin. On the complexity of derivations in propositional calculus. In *Studies in constructive mathematics and mathematical logic, Part II.* Consultants Bureau, New-York-London, 1968.