Polynomial Calculus for MaxSAT

Ilario Bonacina
Polytechnic University of Catalonia, Barcelona, Spain

Maria Luisa Bonet
Polytechnic University of Catalonia, Barcelona, Spain

Jordi Levy
Artificial Intelligence Research Institute, Spanish Research Council (IIIA-CSIC), Barcelona, Spain

Abstract

MaxSAT is the problem of finding an assignment satisfying the maximum number of clauses in a CNF formula. We consider a natural generalization of this problem to generic sets of polynomials and propose a weighted version of Polynomial Calculus to address this problem.

Weighted Polynomial Calculus is a natural generalization of MaxSAT-Resolution and weighted Resolution that manipulates polynomials with coefficients in a finite field and either weights in $\mathbb{N}$ or $\mathbb{Z}$. We show the soundness and completeness of these systems via an algorithmic procedure.

Weighted Polynomial Calculus, with weights in $\mathbb{N}$ and coefficients in $\mathbb{F}_2$, is able to prove efficiently that Tseitin formulas on a connected graph are minimally unsatisfiable. Using weights in $\mathbb{Z}$, it also proves efficiently that the Pigeonhole Principle is minimally unsatisfiable.

2012 ACM Subject Classification Theory of computation → Proof complexity; Theory of computation → Automated reasoning

Keywords and phrases Polynomial Calculus, MaxSAT, Proof systems, Algebraic reasoning

Digital Object Identifier 10.4230/LIPIcs.SAT.2023.5

Funding This work was supported by the Spanish Agencia Estatal de Investigación with the grant PID2019-109137GB-C21/AEI/10.13039/501100011033 and the grant PID2019-109137GB-C22/AEI/10.13039/501100011033.
Ilario Bonacina: Partially supported by the grant IJC2018-035334-I/AEI/10.13039/501100011033.
Maria Luisa Bonet: Partially supported by Simons Institute.
Jordi Levy: Partially supported by Simons Institute.

1 Introduction

The question of whether a set of polynomials $F = \{f_1, \ldots, f_m\}$ is satisfiable – i.e. to know if there exists an assignment of the variables α s.t. $f_i(\alpha) = 0$ for every i – is at the root of algebraic geometry, and it is a natural generalization of SAT, since we can encode CNF formulas as sets of polynomials (over $\{0, 1\}$-valued variables).

The state-of-the-art of practical SAT solving is dominated by CDCL SAT solvers, all of them based on the Resolution proof system [28, 3]. In the last two decades, these solvers have reached remarkable efficiency in industrial SAT instances, but to get further substantial improvements we think it will be necessary to broaden the current paradigm beyond Resolution. Therefore it makes sense to look at the problem from a different point of view using algebraic language and methods to have an impact on solving instances.

Another line of investigation is focusing on encodings to overcome the limitations of CDCL solving. For instance, [21, 5] has shown that the dual-rail encoding allows translating SAT instances into MaxSAT problems. This results in translations of the Pigeonhole Principle with polynomial size proofs using MaxSAT resolution. The same applies to the translation of SAT to Max2SAT using the gadget described in [2]. Moreover, in both cases, general-purpose MaxSAT solvers are able to solve these instances in practice, even though these MaxSAT solvers are not based on MaxSAT resolution.

We have algebraic systems that are stronger than Resolution, and therefore it makes sense to extend those systems to solve MaxSAT problems to see if we can improve on the dual-rail and Max2SAT translations. Moreover, algebraic systems inherently allow more alternative encodings of the problems. For instance, we can use a direct encoding into polynomials, or encode first via a CNF and then translate them into polynomials. These encodings allow us to use algorithms to compute Groebner bases [10, 9, 18], and efficiency maybe gained by these changes. For instance, a direct algebraic encoding, and Groebner-based techniques are useful in practice for coloring [14, 15, 16] and the verification of multiplier circuits [25, 23, 24, 22]. The proof system capturing Groebner-based algorithms is *Polynomial Calculus* (PC) [12], which is a proof system strictly stronger than Resolution. Polynomial Calculus is degree-automatable, in the sense that bounded degree proofs can be found efficiently (in time $n^{O(d)}$, where d is the degree). This is one more reason to extend PC techniques to MaxSAT.

In this paper, we consider the generalization of MaxSAT to the context of polynomials, that is the question of what is the maximum number of polynomials of a given set we are able to simultaneously satisfy. Equivalently, the minimum number of polynomials that we cannot satisfy. In this algebraic context, there are also MaxSAT problems that have natural direct encodings as sets of polynomials, for instance, max-cut or max-coloring.

We define an extension of PC suitable for MaxSAT, i.e. a system that not only is able to prove that a set of polynomials is unsatisfiable but to prove what is the maximum number of polynomials that can be satisfied simultaneously.

Our generalization of PC to a system suitable for MaxSAT is done in a similar way as the adaptation of Resolution to systems suitable for MaxSAT, for instance, MaxSAT-Resolution [7, 8], and weighted Resolution [6]. As weighted Resolution is a system for MaxSAT handling *weighted* clauses, we consider weighted PC, a system handling weighted polynomials. We consider polynomials over *finite* fields. The intuitive reason for this is that, over a finite field $\mathbb{F}_q$ with q elements, we can express $f \neq 0$ as the polynomial equality $f^{q-1} = 1$. We define weighted Polynomial Calculus for polynomials with coefficients in $\mathbb{F}_2$ in Section 3 and in Section 6 we give the definition in the general case.

We call $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ the weighted version of Polynomial Calculus handling weighted polynomials with coefficients in $\mathbb{F}_2$ and weights in $\mathbb{N}$. Intuitively the positive weight of a clause/polynomial is the "penalty" we pay to falsify it. Weighted Resolution has been generalized to $\mathbb{Z}$-weighted Resolution, i.e. weighted resolution but with negative weights [27, 6, 26, 30]. In a similar way, we also define $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}$ as $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ but where we allow negative weights in the proofs. Intuitively the meaning of a weighted clause/polynomial with a negative weight is that it is introduced in the proof as an "assumption" to be justified later, and the negative weight is to keep track of such assumptions yet to be justified.

Connections of weighted Polynomial Calculus with other MaxSAT systems

It is well known that PC (with an appropriate encoding of CNF formulas, the twin variable encoding, see Section 2.2) simulates Resolution. This immediately gives that $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ with twin variables is as strong as $\mathbb{N}$-weighted Resolution and $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}$ is as strong as $\mathbb{Z}$-weighted Resolution (aka Sherali-Adams and Circular Resolution [4, 6]). Pictorially the relations between $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$/$\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}$ and the aforementioned systems can be summarized as in Fig. 1.

None of the systems above are equivalent since $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ and $\mathbb{Z}$-weighted Resolution are incomparable. In one direction Tseitin(G) is easy in $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ while it is hard for $\mathbb{Z}$-weighted Resolution. This follows from the lower bound in [20]. (Tseitin formulas are treated in more detail in Section 5.) In the other direction, the Pigeonhole Principle is easy for $\mathbb{Z}$-weighted

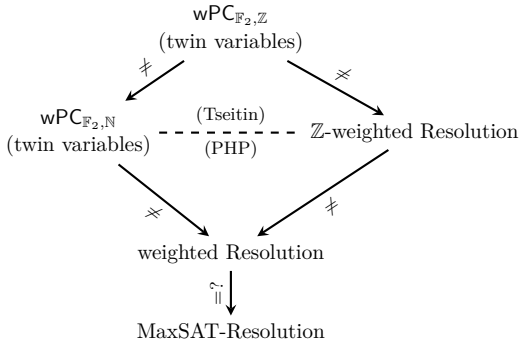

Figure 1 $P \to Q$ means that P is at least as strong as Q. A dashed line means the two systems are incomparable.

Resolution (see for instance [4]) and hence for wPC$_{\mathbb{F}_2,\mathbb{Z}}$, while it is hard for wPC$_{\mathbb{F}_2,\mathbb{N}}$. This follows from the lower bound on the Pigeonhole principle in Polynomial Calculus [29]. Both the Pigeonhole principle and Tseitin have short proofs in wPC$_{\mathbb{F}_2,\mathbb{Z}}$.

We recall that Figure 1 can also be read in the context of propositional proof systems. Indeed, all the MaxSAT systems in Figure 1 can be seen as propositional proof systems, if the weights of the initial clauses/polynomials are *not* part of the input but part of the proof and to refute we just want to derive one instance of the empty clause or the polynomial 1. In this setting weighted Resolution is the same as Resolution and wPC$_{\mathbb{F}_2,\mathbb{N}}$ is the same as PC over $\mathbb{F}_2$.

Analogous simulations as the ones in Fig. 1 hold also for wPC$_{\mathbb{F}_q,\mathbb{N}}$/wPC$_{\mathbb{F}_q,\mathbb{Z}}$.

The main technical contribution of this work is the proof of the completeness of wPC$_{\mathbb{F}_q,\mathbb{N}}$. This is proved in detail for $\mathbb{F}_2$ in Section 4 and we show how to adapt it to the general case in Section 6. The completeness is proved via a saturation process which is a natural generalization of an analogous process used in [7, 8, 1] to prove the completeness of MaxSAT-Resolution. Unlike for MaxSAT-Resolution, we take a more semantic view of the process which makes easier to adapt it to the context of polynomials.

Structure of the paper

Section 2 contains all the necessary preliminaries, in particular, the definition of PC and the extension of MaxSAT to polynomials. In Section 3, we give the formal definition of wPC$_{\mathbb{F}_2,\mathbb{N}}$ and wPC$_{\mathbb{F}_2,\mathbb{Z}}$. Section 4 contains the completeness of wPC$_{\mathbb{F}_2,\mathbb{N}}$. In Section 5, we give an application of the saturation process to Tseitin formulas. Section 6 shows how to generalize the definition of wPC$_{\mathbb{F}_2,\mathbb{N}}$ and wPC$_{\mathbb{F}_2,\mathbb{Z}}$ from $\mathbb{F}_2$ to a generic finite field. Finally, in Section 7, we give some concluding remarks.

2 Preliminaries

For $n \in \mathbb{N}$, let $[n] = \{1, \dots, n\}$. In general, we use capital letters to denote (multi-)sets.

2.1 Polynomial Calculus

Let $\mathbb{F}_q$ be a finite field with q elements (it exists whenever $q = p^k$ for some prime p and integer k). For most of this paper, we focus on $q = 2$, i.e. on the field with two elements $0, 1$. Given a set of variables X, with $\mathbb{F}_q[X]$ we denote the ring of multivariate polynomials with coefficients in $\mathbb{F}_q$ and variables in X.

We denote polynomials using the letters f, g, while we use Greek letters to denote assignments. An assignment is a function $\alpha\colon X \to \mathbb{F}_q$ and for a polynomial $f \in \mathbb{F}_q[X]$, $f(\alpha)$ is the *evaluation* of f in α: the element of $\mathbb{F}_q$ resulting from substituting each variable x in f with $\alpha(x)$ and simplifying the resulting expression. If $f(\alpha) = 0$ we say that α *satisfies* f. The polynomial 1 represents the unsatisfiable polynomial (the equivalent to the empty clause in SAT).

Next, we define the algebraic proof system *Polynomial Calculus*, originally introduced by Clegg et al. [12]. Even though the system can be defined for any field (or even rings, see for instance [11]), in this paper we only consider finite fields.

Polynomial Calculus over $\mathbb{F}_q$ ($\mathsf{PC}_{\mathbb{F}_q}$) is a proof system that handles polynomials in $\mathbb{F}_q[X]$. A derivation in $\mathsf{PC}_{\mathbb{F}_q}$ of a polynomial f from a set of polynomials F is a sequence of polynomials $f_1, \ldots, f_m$, where $f_m = f$, and for each i either $f_i \in F$, or $f_i = gf_j$ for some $g \in \mathbb{F}_q[X]$ and $j < i$, or $f_i = f_j + f_k$ for some $j, k < i$. A *refutation* is a derivation of the polynomial 1, and the *size* of a derivation is the total number of bits needed to express it.

Sometimes, the inference rules of $\mathsf{PC}_{\mathbb{F}_q}$ are given as

$$\frac{f \quad g}{f+g}\,, \qquad \frac{f}{\alpha f} \qquad \text{and} \qquad \frac{f}{xf}$$

for all $f, g \in \mathbb{F}_q[X]$, $\alpha \in \mathbb{F}_q$, and $x \in X$. This will just give a polynomial increase in the size of the derivations (hence it is p-equivalent to our presentation of $\mathsf{PC}_{\mathbb{F}_q}$). $\mathsf{PC}_{\mathbb{F}_q}$ – together with an encoding of formulas into polynomials – is a Cook-Reckhow propositional proof system [13].

2.2 From formulas in CNF to polynomials

A *clause* C is a set of literals, i.e. Boolean variables x_i or negated Boolean variables $\neg x_i$ from a given set of variables $\{x_1, \ldots, x_n\}$. A CNF formula is a set of clauses, and a k-CNF is a CNF where each clause has at most k literals. An assignment $\alpha\colon \{x_1, \ldots, x_n\} \to \{0, 1\}$ satisfies a clause if it maps at least a literal to 1, where $\alpha(\neg x_i) := 1 - \alpha(x_i)$. An assignment satisfies a CNF formula if it satisfies all the clauses in it.

To encode CNF formulas into sets of polynomials that could be refuted in $\mathsf{PC}_{\mathbb{F}_q}$, we use the following encoding with *twin variables*. We call *twin variables* the set of variables $X = \{x_1, \ldots, x_n, \bar{x}_1, \ldots, \bar{x}_n\}$. The intended meaning of $\bar{x}_i$ is $1 - x_i$.

For every clause $C = \{x_i : i \in I\} \cup \{\neg x_j : j \in J\}$, we associate the monomial $M(C) = \prod_{i \in I} \bar{x}_i \prod_{j \in J} x_j$ in the twin variables X. Then a set of clauses $\{C_1, \ldots, C_m\}$ is encoded as

$$\{M(C_1),\ \ldots,\ M(C_m)\} \cup \{x_i^2 - x_i,\ x_i + \bar{x}_i - 1\ \ : i \in [n]\}\ .$$

Any assignment $\alpha\colon \{x_1, \ldots, x_n\} \to \{0, 1\}$ can be extended to an assignment $\alpha' : X \to \{0, 1\}$, where for each $i \in [n]$, $\alpha'(x_i) = \alpha(x_i)$ and $\alpha'(\bar{x}_i) = 1 - \alpha(x_i)$. Then α satisfies a CNF formula (i.e. α maps all the clauses to 1) if and only if α' satisfies the polynomial encoding of F (i.e. α' is a common solution of the polynomials).

With this encoding of CNF formulas into polynomials, it is well-known that for every q, $\mathsf{PC}_{\mathbb{F}_q}$ p-simulates Resolution and indeed the p-simulation is strict [11]. For example, Tseitin formulas (see Section 5) have polynomial size $\mathsf{PC}_{\mathbb{F}_2}$ refutations while they require exponential size in Resolution [32]. Notice that the variables $\bar{x}_i$s are not strictly needed for the encoding (they could be eliminated just by substituting $1 - x_i$ for each occurrence of $\bar{x}_i$), but $\mathsf{PC}_{\mathbb{F}_q}$ with this alternative encoding does not p-simulate Resolution, not even on k-CNFs [17]. With different encodings of CNF formulas, for instance, using $\{\pm 1\}$-valued variables, it is open whether $\mathsf{PC}_{\mathbb{F}_q}$ simulates Resolution (see [31] for lower bounds on $\mathsf{PC}_{\mathbb{Q}}$ with the ± 1-variables).

2.3 MaxSAT on sets of polynomials

Let X be a generic set of variables. To define partial weighted MaxSAT, we distinguish between hard and soft clauses. The hard clauses need to be satisfied, while the soft ones consist of a clause and a weight (a number in $\mathbb{N}$). The weight of a soft clause is the cost of falsifying it. Given a set of soft clauses F and a set of hard ones H, *Weighted Partial MaxSAT* is the problem of finding an assignment to the variables X that satisfies the hard clauses H, and that minimizes the cost of the falsified soft clauses F.

In this paper, we generalize partial weighted MaxSAT to arbitrary polynomials in $\mathbb{F}_q[X]$. The generalization of the *hard* constraints of MaxSAT is some set of polynomials $H \subset \mathbb{F}_q[X]$, while the generalization of the *soft* constraints is a multi-set of the form

$$F = \{[\, f_1 \,,\, w_1 \,], \ldots, [\, f_m \,,\, w_m \,]\} \,,$$

where $f_i \in \mathbb{F}_q[X]$ and $w_i \in \mathbb{N}$. A pair $[\, f \,,\, w \,]$ where $f \in \mathbb{F}_q[X]$ and $w \in \mathbb{Z}$ is a *weighted polynomial*. The weight w informally measures the "importance" we give to satisfying the polynomial f. In this context, we are interested in assignments α that minimize the weight of the falsified soft polynomials in F, and satisfy all the polynomials in H.

▶ **Definition 2.1** (H-compatible assignment). *Let $H \subseteq \mathbb{F}_q[X]$. An assignment $\alpha \colon X \to \mathbb{F}_q$ is* H-compatible *if for every $h \in H$, $h(\alpha) = 0$.*

Now, for each assignment $\alpha \colon X \to \mathbb{F}_q$, we measure how close it is to satisfying all the polynomials in F, and we do this by defining its *cost* as the sum of the weights of the polynomials in F not satisfied by α. Therefore, the cost of the assignment α on F is

$$\mathrm{cost}(\alpha, F) = \sum_{i \in [m]} w_i \chi_i(\alpha) \,, \tag{1}$$

where $\chi_i(\alpha)$ is 1 if $f_i(\alpha) \neq 0$, and 0 otherwise. Finally, to solve a partial weighted MaxSAT problem, we want the minimum value of $\mathrm{cost}(\alpha, F)$ for any H-compatible assignment α, i.e.

$$\mathrm{cost}_H(F) = \min_{\alpha \ H\text{-compatible}} \mathrm{cost}(\alpha, F) \,. \tag{2}$$

If $H = \emptyset$, we denote $\mathrm{cost}_H(F)$ simply as $\mathrm{cost}(F)$. Of course, if F is satisfiable using a H-compatible assignment, then $\mathrm{cost}_H(F) = 0$.

Notice that, the polynomials in H and the weighted polynomials in F could come from the translation of a partial weighted MaxSAT instance. However, we cannot assume this is always the case. Moreover, the polynomials in H could be used to enforce specific types of assignments.

▶ **Example 2.2** (Boolean axioms). If $H = \{x^2 - x : x \in X\}$, the H-compatible assignments are all the functions $\alpha \colon X \to \{0, 1\}$. We refer to this H as *Boolean axioms*. If we are over $\mathbb{F}_2$ then, equivalently, H could be taken as $\emptyset$.

In the case of twin variables $X = \{x_1, \ldots, x_n, \bar{x}_1, \ldots, \bar{x}_n\}$ and the Boolean axioms $H = \{x_i^2 - x_i, x_i + \bar{x}_i - 1 : i \in [n]\}$, the H-compatible assignments are all the functions $\alpha \colon X \to \{0, 1\}$ satisfying the additional property that for each $i \in [n]$, $\alpha(x_i) + \alpha(\bar{x}_i) = 1$. We refer to this case as *Boolean axioms with twin variables*. Similarly as before, if we are over $\mathbb{F}_2$, then, equivalently, H could be taken as $\{x_i + \bar{x}_i - 1 : i \in [n]\}$.

Sometimes a direct encoding with polynomials not coming from CNF formulas is more natural. For instance, this is the case of max-cut.

▶ **Example 2.3** (max-cut). Given a graph $G = (V, E)$ consider $X = \{x_v : v \in V\}$ and let $F = \{[\, x_{v_1} + x_{v_2} + 1 \,,\, 1 \,] : (v_1, v_2) \in E\} \subseteq \mathbb{F}_2[X]$. Finding $\mathrm{cost}(F)$ is equivalent to finding a max-cut in G. This codification could be easily generalized to weighted max-cut.

3 Polynomial Calculus for MaxSAT

We first define Polynomial Calculus for MaxSAT in the special case of polynomials with coefficients in $\mathbb{F}_2$ (the general case is in Section 6). Recall that $\mathbb{F}_2$ is the finite field with 2 elements 0 and 1 (this field is unique up to isomorphism), in particular for each element of $a \in \mathbb{F}_2$, $a^2 = a$ and $2 \cdot a = a + a = 0$.

The initial instance consists of a multi-sets of weighted polynomials, i.e. pairs $[\, f \,,\, w \,]$ with $f \in \mathbb{F}_2[X]$ and $w \in \mathbb{N}$, and a set of hard polynomials H. We define $\mathbb{Z}$-*weighted Polynomial Calculus* ($\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}$), an inference system that handles weighted polynomials with weights in $\mathbb{Z}$, and $\mathbb{N}$-*weighted Polynomial Calculus* ($\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$), an inference system that handles weighted polynomials with weights in $\mathbb{N}$. Both systems use the same set of inference rules. The formal definition of $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}/\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ is Definition 3.3, but we discuss first the inference rules of the system. They are two types: structural rules (the FOLD, UNFOLD and the H-SIMPLIFICATION), and proper inference rules (SUM and SPLIT).

To have a sound proof system in the context of partial weighted MaxSAT, we use the inference rules as *substitution* rules, that is, when applied, these rules *replace* the premises with the conclusions. In this context, a substitution rule is *sound* if, for every assignment α, the cost of the set of premises on α equals the cost of the conclusions on α.

Fold-unfold. Let F, G be two multi-sets of weighted polynomials, we say that F and G are *fold-unfold equivalent* ($F \approx G$) if there is a sequence of multi-sets of weighted polynomials starting with F and ending with G where from one multi-set to the next, one of the following substitution rules is applied

$$\frac{[\, f \,,\, u \,] \quad [\, f \,,\, w \,]}{[\, f \,,\, u + w \,]}\ (\text{FOLD}) \qquad \frac{[\, f \,,\, u + w \,]}{[\, f \,,\, u \,] \quad [\, f \,,\, w \,]}\ (\text{UNFOLD})$$

$$\frac{[\, f \,,\, 0 \,]}{}\ (0\text{-FOLD}) \qquad \frac{}{[\, f \,,\, 0 \,]}\ (0\text{-UNFOLD})$$

where $f \in \mathbb{F}_2[X]$, and $w, u \in \mathbb{Z}$.

▶ **Example 3.1.** For instance, $\{[\, f \,,\, 0 \,]\} \approx \emptyset$ and $\{[\, f \,,\, 2 \,]\} \approx \{[\, f \,,\, 1 \,],\ [\, f \,,\, 1 \,]\}$.

It is immediate to see that the fold-unfold equivalence rules are sound. Notice that this fold-unfold equivalence is similar to the fold-unfold equivalence used in [6] in the context of weighted clauses and weighted Resolution.

H-equivalence. In $\mathbb{F}_2[X]$, the polynomials $x^2 - x$ and 0 are two distinct polynomials, but since they always evaluate to 0, we want to identify them. Moreover, given a set of hard constraints H, we are only interested in H-compatible assignments, hence we want to identify two polynomials f, g such that for every H-compatible assignment α, $f(\alpha) = g(\alpha)$. This can be seen as having a EQUIVALENCE rule of the form

$$\frac{[\, f \,,\, w \,]}{[\, g \,,\, w \,]}\ (H\text{-EQUIVALENCE})$$

where $f, g \in \mathbb{F}_2[X]$ and $w \in \mathbb{Z}$ are such that for every H-compatible assignment $\alpha \colon X \to \mathbb{F}_2$, $f(\alpha) = g(\alpha)$. If f and g are H-EQUIVALENT we write $f \equiv_H g$ (when H is clear from the context we simply write $f \equiv g$). In particular, for every H, $f^2 \equiv_H f$.

Notice that, by definition, if $f \equiv_H g$ then the cost is preserved on every H-compatible α, hence the rule is sound on H-compatible αs. To efficiently check whether $f \equiv_H g$ might be problematic, depending on H.

▶ Remark 3.2. The H-equivalence could be checked efficiently for $H = \emptyset$, just by looking at the multilinearization of the polynomials. The multilinearization of a polynomial f, $\mathrm{mul}(f)$ is the unique multilinear polynomial H-equivalent to f.

In the case of polynomials coming from the direct translation of CNF formulas, we use the H-equivalence for twin variables and H being the Boolean axioms for twin variables. In this case, the H-equivalence can also be checked efficiently [19, section 4.3 and Theorem 4.4].

Sum and split. Apart from the previous structural rules, in $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ and $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}$ we have the following inference rules:

$$\frac{[\,f\,,\,w\,]}{[\,fg\,,\,w\,]\quad[\,f(g+1)\,,\,w\,]}\ (\textsc{split}) \qquad \frac{[\,f\,,\,w\,]\quad[\,g\,,\,w\,]}{[\,f+g\,,\,w\,]\quad[\,fg\,,\,2w\,]}\ (\textsc{sum}) \tag{3}$$

for all $f, g \in \mathbb{F}_2[X]$ and $w \in \mathbb{Z}$.

Notice that the previous rules are *sound*. For the SPLIT rule, if $f(\alpha) = 0$ then both the conclusions are 0, but if $f(\alpha) = 1$, then exactly one of the conclusions is 1. For the SUM rule, the argument by cases is analogous. The case that justifies the weight of $2w$ for the polynomial fg in the conclusion is when $f(\alpha) = 1$ and $g(\alpha) = 1$. In this case $f(\alpha)+g(\alpha) = 0$ and $f(\alpha)g(\alpha) = 1$, hence the weight of the conclusion fg should equal the sum of the weights of both premises, which is two.

Formally the definition of $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}$/$\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ is the following.

▶ **Definition 3.3** ($\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}$, $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$). *Given a multi-set of weighted polynomials F and a set of hard constraints H, a $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}$ derivation of a weighted polynomial $[\,f\,,\,w\,]$ from F and H is a sequence of multi-sets $L_0, \ldots, L_\ell$ s.t.*

1. $L_0 = F$,
2. $[\,f\,,\,w\,] \in L_\ell$ *and all the other weighted polynomials* $[\,f'\,,\,w'\,] \in L_\ell$ *have* $w' \in \mathbb{N}$, *and*
3. *for each* $i > 0$ *either* $L_i \approx L_{i-1}$ *or* L_i *is the result of an application of the* SPLIT/SUM/H-EQUIVALENCE *rule as a substitution rule on* L_{i-1}.

The system $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ is the restriction of $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}$ where all weights are natural numbers. The size *of a $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}$/$\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ derivation $L_0, \ldots, L_\ell$ is the total number of occurrences of symbols in $L_0, \ldots, L_\ell$.*

To clarify the definition, we give an example of derivation in $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$.

▶ **Example 3.4** (Example 2.3 cont.). In Fig. 2 we show a $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$-derivation of $[\,1\,,\,2\,]$ from the set of polynomials we saw in Example 2.3 in the case of G being the clique on 4 vertices. That is the weighted polynomials $[\,x+y+1\,,\,1\,]$, $[\,x+z+1\,,\,1\,]$, $[\,x+t+1\,,\,1\,]$, $[\,y+z+1\,,\,1\,]$, $[\,y+t+1\,,\,1\,]$, $[\,z+t+1\,,\,1\,]$. In this derivation, the polynomials that are just copied from one multiset to the next are substituted with a $\bullet$. From one multiset to the next we applied multiple rules in parallel. The horizontal lines are just a visual help to visualize the multisets. Notice that we have H-equivalences (for $H = \emptyset$) applied implicitly. For instance, some SUM only have one consequence since the other is equivalent to 0. This example shows that to obtain $[\,1\,,\,2\,]$ it is important to use both consequences of a SUM.

We prove now the *soundness* of $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}$.

▶ **Theorem 3.5** (soundness). *Given $F = \{[\,f_1\,,\,w_1\,], \ldots, [\,f_m\,,\,w_m\,]\}$ where $f_i \in \mathbb{F}_2[X]$ and a set of polynomials $H \subseteq \mathbb{F}_2[X]$, if there is a $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}$ derivation of $[\,1\,,\,w\,]$ from F (and H as hard constraints), then $\mathrm{cost}_H(F) \geq w$.*

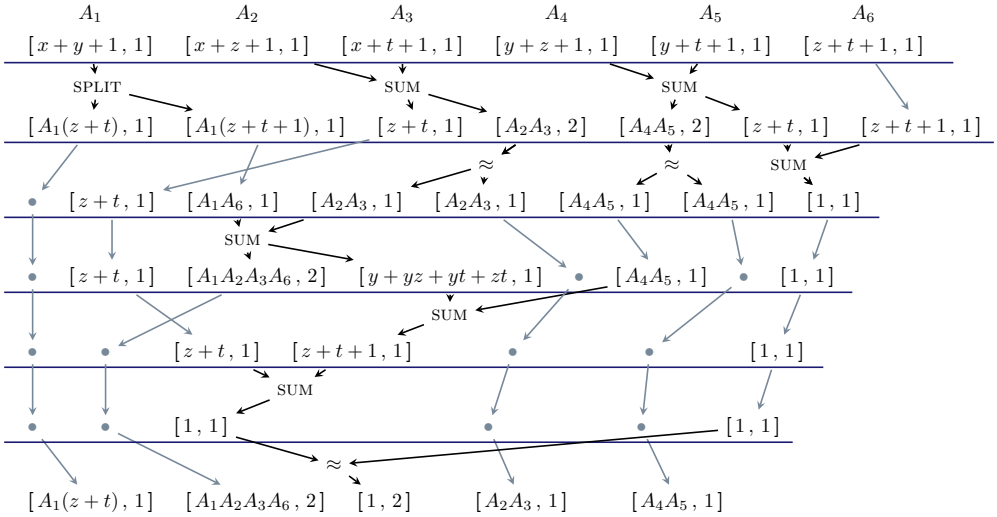

Figure 2 A $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ derivation of $[1,2]$ from the axioms of max-cut on a clique of 4 vertices: $[x_1+x_2+1,1]$, $[x_1+x_3+1,1]$, $[x_1+x_4+1,1]$, $[x_2+x_3+1,1]$, $[x_2+x_4+1,1]$, $[x_3+x_4+1,1]$.

Proof. Let $L_0, L_1, L_2, \ldots, L_s$ be a $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}$ derivation of $(1;w)$, i.e. L_s contains $[1,w]$, $L_0 = F$ and each L_{i+1} is obtained from L_i applying the SPLIT, the SUM substitution rules, the fold-unfold equivalence or the H-SIMPLIFICATION. We have to show that $\mathrm{cost}_H(F) \geq w$. We have that $\mathrm{cost}_H(L_s) \geq w$ since $[1,w] \in L_s$ and all the other weighted polynomials in L_s have non-negative weights. Hence, to prove the statement is enough to show that for each i, $\mathrm{cost}_H(L_{i+1}) = \mathrm{cost}_H(L_i)$. We prove something slightly stronger, that for each H-consistent $\alpha\colon X \to \mathbb{F}_2$, $\mathrm{cost}(\alpha, L_{i+1}) = \mathrm{cost}(\alpha, L_i)$. This follows immediately from the comments we already made on the soundness of the various rules. ◀

We conclude this section with an observation on the SPLIT and SUM rules in $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}$. Using weights in $\mathbb{Z}$, one of them is always redundant, unlike the case of weights in $\mathbb{N}$ where both are necessary. To simulate the SPLIT rule using the SUM rule using weights in $\mathbb{Z}$ we can do the following:

$$\dfrac{\dfrac{\dfrac{[f,w]}{[f,w]\quad[fg,w]\quad[fg,-w]\quad[f(g+1),w]\quad[f(g+1),-w]}\;\approx}{[f,w]\quad[fg,w]\quad[f(g+1),w]\quad[fg+f(g+1),-w]\quad[f^2g(g+1),-2w]}\;\text{SUM}}{[fg,w]\quad[f(g+1),w]}\;\approx\ \&\ \equiv$$

In a similar way, we can also simulate the SUM using the SPLIT rule using weights in $\mathbb{Z}$. (The proof of this will appear in the final version of this paper.)

4 Completeness

We show the completeness of $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$, that is the converse of Theorem 3.5. For simplicity, we focus on the Boolean axioms as hard constraints, that is $H = \emptyset$ or, in the case of twin variables $\{x_1, \ldots, x_n, \bar{x}_1, \ldots, \bar{x}_n\}$, $H = \{x_i + \bar{x}_i - 1 : i \in [n]\}$.

▶ **Theorem 4.1** (completeness for Boolean axioms). *Given F a set of weighted polynomials over $\mathbb{F}_2[X]$, there is a* $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ *derivation of* $[\,1\,,\,\mathrm{cost}_H(F)\,]$ *from F and the set of Boolean axioms as hard constraints H.*

Our proof generalizes the *saturation* process from [8] and gives an algorithm to find $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$-derivations of $[\,1\,,\,\mathrm{cost}_H(F)\,]$. We five an example of the construction in Section 5. Clearly the completeness for $H = \emptyset$ implies the completeness for $H = \{x_i + \bar{x}_i - 1 : i \in [n]\}$. For instance, just removing the twin variables, that is substituting each variable $\bar{x}_i$ with $1 - x_i$. We show a saturation process that adapts to this context without removing the twin variables.

Our construction shows indeed something stronger, that $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ is still complete even if we restrict the SPLIT rule of $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ to be of the form

$$\frac{[\,f\,,\,w\,]}{[\,ff_{x=0}f_{x=1}\,,\,w\,] \quad [\,f(f_{x=0}f_{x=1}+1)\,,\,w\,]}\,,$$

where x is some variable and $f_{x=0}$ is the polynomial resulting from the restriction of f mapping x to 0, and analogously for $f_{x=1}$. In the case of twin variables, for $f_{x=0}$ we also map $\bar{x}$ to 1, to be consistent with the Boolean axioms, and analogously for $f_{x=1}$.

Recall that for polynomials $f, g \in \mathbb{F}_2[X]$, let $f \equiv g$ if for every H-compatible assignment α, $f(\alpha) = g(\alpha)$. It is immediate to see that $ff_{x=0}f_{x=1} \equiv f_{x=0}f_{x=1}$ since for every value $a \in \mathbb{F}_2$, $a^2 = a$. Therefore, if $f \not\equiv f_{x=0}f_{x=1}$ and $f_{x=0}f_{x=1} \not\equiv 0$, the special case of the SPLIT rule above allows to infer from f some new polynomials and one of them ($f_{x=0}f_{x=1}$) without the variable x.

▶ **Definition 4.2.** *We say that a polynomial f* depends *on a variable x if for every polynomial g not containing x (and also $\bar{x}$ in the case of twin variables), $f \not\equiv g$.*

Notice that, the following are equivalent:

- f depends on x,
- For $H = \emptyset$, the multilinearization of f is a polynomial $xf_1 + f_0$ with f_1, f_0 not containing x and $f_1 \not\equiv 0$. For the twin variables and $H = \{x_i + \bar{x}_i - 1 : i \in [n]\}$, f is equivalent to a multilinear polynomial of the form $xf_1 + \bar{x}f_1' + f_0$, with f_1, f_1', f_0 not containing $x, \bar{x}$, and $f_1 \not\equiv f_1'$.
- $f \not\equiv f_{x=0}f_{x=1}$.

(The proof of this will appear in the final version of this paper.) The reason we give these equivalences is that the third condition makes easier to generalize the whole construction to arbitrary finite fields.

The main concept used to show the completeness of $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ is the notion of set of polynomials *saturated* w.r.t. a variable.

▶ **Definition 4.3** (x-saturated set). *Let $x \in X$ and S be a set of weighted polynomials. The set S is* x-saturated *if every H-compatible assignment $\alpha\colon X \to \mathbb{F}_2$ can be modified in x to a H-compatible assignment satisfying all weighted polynomials in S that depend on x.*

Notice that, if S is x-saturated then the subset of polynomials in S depending on x is satisfiable, but the converse is not true. For instance, $\{[\,x + y\,,\,1\,],\ [\,x + z\,,\,1\,]\}$ is clearly satisfiable but it is not saturated w.r.t. to x since we cannot extend the assignment $y = 0, z = 1$ to satisfy both polynomials.

Next, we give a procedure to x-saturate a set of weighted polynomials S. Recall that we focus on H being the Boolean axioms. Informally, the procedure to obtain the x-saturation consists of applying the SPLIT rule to a polynomial or the SUM of two polynomials, as long

as the application of these rules generates polynomials that don't contain the variable x, and are not equivalent to 0. The procedure is applied as long as it is possible, and we will see that it finishes in a finite number of steps and when it terminates the generated set must be x-saturated.

▶ **Lemma 4.4.** *In the context of H the Boolean axioms, for every set of weighted polynomials S and every variable x, there is a $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ derivation of a set of polynomials S' which is x-saturated.*

Proof. For a polynomial f, recall that $f_{x=0}$ is the evaluation of f in $x = 0$ and, in the case of twin variables, the restriction also sets $\bar{x} = 1$ (resp. for $f_{x=1}$).

Suppose we have a set of weighted polynomials S and a variable x. We construct a sequence of weighted polynomials $S_0, S_1, \dots$ to find the saturation. We start with $S_0 = S$, then we want S_{i+1} to be derivable from S_i using the rules of $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$, and moreover, in S_{i+1} we added some new polynomial non-dependent on x. The way to obtain S_{i+1} from S_i is the following. For each $i \geq 0$, if there is an $[\, f \,,\, w \,] \in S_i$ depending on x and s.t. $f_{x=0}f_{x=1} \not\equiv 0$, non-deterministically choose one of such $[\, f \,,\, w \,]$ and let

$$S_{i+1} = (S_i \setminus \{[\, f \,,\, w \,]\}) \cup \{[\, f_{x=0}f_{x=1} \,,\, w \,],\ [\, f_{x=0}f_{x=1} + f \,,\, w \,]\} \ .$$

The derivation of S_{i+1} from S_i, by substituting $[\, f \,,\, w \,]$ with the weighted polynomials $[\, f_{x=0}f_{x=1} \,,\, w \,]$ and $[\, f_{x=0}f_{x=1} + f \,,\, w \,]$, is justified by:

$$\dfrac{\dfrac{[\, f \,,\, w \,]}{[\, ff_{x=0}f_{x=1} \,,\, w \,] \quad [\, f(f_{x=0}f_{x=1} + 1) \,,\, w \,]}\ \equiv}{[\, f_{x=0}f_{x=1} \,,\, w \,] \quad [\, f_{x=0}f_{x=1} + f \,,\, w \,]}\ \textsc{split}$$

where the last $\equiv$ holds since $ff_{x=0}f_{x=1} \equiv f_{x=0}f_{x=1}$. Notice that, with this substitution, we have obtained the weighted polynomial $[\, f_{x=0}f_{x=1} \,,\, w \,]$ where the variable x doesn't appear (and hence clearly not depending on x) and it is not equivalent to 0 since the condition to obtain S_{i+1} is that $f_{x=0}f_{x=1} \not\equiv 0$. We used the assumption that f depends on x to ensure the polynomial $f_{x=0}f_{x=1}$ in the conclusions is new and it is not equivalent to the polynomial f in the premises.

If there are $[\, f \,,\, w \,], [\, g \,,\, w' \,] \in S_i$ depending on x, with $f \not\equiv g$,

$$(f_{x=0} + g_{x=0})(f_{x=1} + g_{x=1}) \not\equiv 0 \ ,$$

and $w' \geq w > 0$, non-deterministically choose two of them. First substitute $[\, g \,,\, w' \,]$ by $[\, g \,,\, w \,]$ and $[\, g \,,\, w' - w \,]$, and then let

$$\begin{aligned} S_{i+1} = (S_i \setminus \{[\, f \,,\, w \,],\ [\, g \,,\, w \,]\}) \cup \{&[\, (f+g)_{x=0}(f+g)_{x=1} \,,\, w \,],\ [\, fg \,,\, 2w \,], \\ &[\, (f+g)_{x=0}(f+g)_{x=1} + f + g \,,\, w \,]\} \ . \end{aligned}$$

We can obtain S_{i+1} from S_i using first the SUM rule to infer $[\, f + g \,,\, w \,]$ and $[\, fg \,,\, 2w \,]$ and then use the SPLIT rule on $[\, f + g \,,\, w \,]$ as we did in the previous case on a single polynomial.

$$\dfrac{\dfrac{[\, f \,,\, w \,] \quad [\, g \,,\, w \,]}{[\, f + g \,,\, w \,] \quad [\, fg \,,\, 2w \,]}\ \textsc{sum}}{[\, (f+g)_{x=0}(f+g)_{x=1} \,,\, w \,] \quad [\, (f+g)_{x=0}(f+g)_{x=1} + f + g \,,\, w \,] \quad [\, fg \,,\, 2w \,]}\ \textsc{split} \ \&\ \equiv$$

Doing this substitution we obtain a polynomial where the variable x doesn't appear and it is not equivalent to 0 since the condition to obtain S_{i+1} is that $(f_{x=0} + g_{x=0})(f_{x=1} + g_{x=1}) \not\equiv 0$.

If we cannot transform S_i to S_{i+1} in neither of the two ways we stop the process.

This sequence of transformations must be finite and the last element S_ℓ will be x-saturated. The process must be finite since otherwise the new sequence given by

$$\sigma(i) = \sum_{\substack{\alpha\colon X\to\mathbb{F}_2 \\ H\text{-compatible}}} \mathrm{cost}(\alpha, S_i^+)$$

for S_i^+ the part of S_i depending on x, would give a sequence of strictly decreasing natural numbers. Indeed, all the $\sigma(i)$s are natural numbers because we are using the rules of $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$, so, no negative weight could appear in any S_i and $\mathrm{cost}(\alpha, S_i^+) \geq 0$. To show that $\sigma(i+1) < \sigma(i)$ it is sufficient to notice that, by the soundness of $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$, for every H-compatible $\alpha\colon X \to \mathbb{F}_2$,

$$\mathrm{cost}(\alpha, S_{i+1}) = \mathrm{cost}(\alpha, S_i) \ ,$$

which implies that

$$\mathrm{cost}(\alpha, S_{i+1}^+) + \mathrm{cost}(\alpha, \{[\, h \,,\, w \,]\}) = \mathrm{cost}(\alpha, S_i^+) \ ,$$

for some polynomial $h \not\equiv 0$, not depending the variable x and with $w > 0$. Hence

$$\sigma(i+1) + \sum_{\substack{\alpha\colon X\to\mathbb{F}_2 \\ H\text{-compatible}}} \mathrm{cost}(\alpha, \{[\, h \,,\, w \,]\}) = \sigma(i) \ ,$$

and

$$\sum_{\substack{\alpha\colon X\to\mathbb{F}_2 \\ H\text{-compatible}}} \mathrm{cost}(\alpha, \{[h, w]\}) > 0$$

because $h \not\equiv 0$ and $w > 0$. Therefore, $\sigma(i+1) < \sigma(i)$. And the sequence must be finite.

Now, S_ℓ, the last set of the sequence, must be x-saturated. Suppose, towards a contradiction, that both α_0, i.e. α modified mapping $x \mapsto 0$, and α_1, i.e. α modified mapping $x \mapsto 1$, falsify some polynomials in S_ℓ depending on x. (In the case of twin variables α_0 also sets $\bar{x} \mapsto 1$ and α_1 also sets $\bar{x} \mapsto 0$.) Let such polynomials resp. be f, g. That is we have $f_{x=0}(\alpha) \neq 0$, and $g_{x=1}(\alpha) \neq 0$. Since S_ℓ is the last element of the previous process we must have that $f_{x=0}f_{x=1} \equiv 0$ and $g_{x=0}g_{x=1} \equiv 0$. Hence it must be that $f_{x=1}(\alpha) = 0$, and $g_{x=0}(\alpha) = 0$. In particular, f and g are two non-equivalent polynomials. Then, again by the assumption on S_ℓ being the last element of the process, we must also have that $(f+g)_{x=0}(f+g)_{x=1} \equiv 0$, but this is not possible since

$$(f+g)(\alpha_0) = (f+g)_{x=0}(\alpha) = f_{x=0}(\alpha) + g_{x=0}(\alpha) \neq 0$$

and similarly $f_{x=1}(\alpha) + g_{x=1}(\alpha) \neq 0$. ◀

Notice that in the proof of the previous lemma, there are many alternative ways to introduce new polynomials not depending on x in each step of the sequence $S_1, S_2, \ldots$. The one we chose has the property that we only use a special form of the SPLIT rule:

$$\frac{[\, f \,,\, w \,]}{[\, f_{x=0}f_{x=1} \,,\, w \,] \quad [\, f_{x=0}f_{x=1} + f \,,\, w \,]} \ .$$

Moreover, we keep the number and the degree of the polynomials we introduce at each step lower than other alternative choices. For instance, given multilinear polynomials $f = xf_1 + f_0$ and $g = xg_1 + g_0$ depending on x, we could have done two SPLIT to obtain $g_1 f$ and $f_1 g$ and

then SUM them to obtain $f_0g_1 + f_1g_0$ (and hence introducing also $[\, f_1g_1fg\,,\, 2\,]$). This would have resulted in the introduction of a larger number and higher degree polynomials compared to the construction we gave.

We now show how to obtain the completeness, under the assumption that the saturation can be computed in $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$. Essentially iterating the saturation on all the variables one by one.

▶ **Lemma 4.5.** *Let $H \subset \mathbb{F}_2[X]$. If for every set of weighted polynomials S and every variable x, there is a $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ derivation of a set of polynomials S' which is x-saturated, then for every set of weighted polynomials F over $\mathbb{F}_2[X]$ there exists a $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$-derivation of $[\,1\,,\, \mathrm{cost}_H(F)\,]$ from F and the hard constraints H.*

Proof. Let $X = \{x_1, \ldots, x_n\}$. First saturate F w.r.t. x_1. By hypothesis, from F in $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ we can derive a x_1-saturated set S_1. Let $S_1 = S_1^+ \cup S_1^-$, where S_1^+ is the part of S_1 depending on x_1 and S_1^- the part of S_1 not depending on x_1.

Saturate S_1^- w.r.t. x_2. Again, by assumption, from S_1^- we can derive in $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ a x_2-saturated set S_2. This gives a decomposition $S_2 = S_2^+ \cup S_2^-$, where S_2^- are the weighted polynomials in S_2 not depending on x_2 (and x_1). Continuing saturating by all the variables of X one by one we arrive at a set S_n, where the weighted polynomials in S_n^- are just constants, i.e. $S_n^- \approx \{[\,1\,,\, w\,]\}$ for some $w \in \mathbb{N}$.

To show that $w = \mathrm{cost}_H(F)$ it is enough to show that $\bigcup_{j\in[n]} S_j^+$ is satisfiable by a H-compatible assignment. Let $\alpha\colon X \to \mathbb{F}_2$ be an arbitrary H-compatible assignment. Since S_n is x_n-saturated there is a way to modify α in x_n to get a H-compatible assignment satisfying all S_n^+. Let this assignment be α_n. Suppose we obtained a H-compatible assignment α_i satisfying $\bigcup_{j\geq i} S_j^+$, since S_{i-1} is x_{i-1}-saturated, there is a way to modify α_i in x_{i-1} to satisfy all S_{i-1}^+. Let this assignment be α_{i-1}. Since the polynomials in $\bigcup_{j\geq i} S_j^+$ only contained the variables $x_i \ldots, x_n$, the assignment α_{i-1} continues to satisfy $\bigcup_{j\geq i} S_j^+$. We continue this way until we get an assignment α_1 satisfying all $\bigcup_{j\in[n]} S_j^+$. Thus proving that it must have been that $w = \mathrm{cost}_H(F)$. ◀

Notice that the previous lemma does not require H to be the Boolean axioms and it is completely general.

Now, it is immediate to prove the completeness of $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ (Theorem 4.1). Indeed, by Lemma 4.4, for every set of weighted polynomials S and every variable x, there is a $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ derivation of a set of polynomials S' which is x-saturated. Then, by Lemma 4.5, there is a $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ derivation of $[\,1\,,\, \mathrm{cost}_H(F)\,]$ from F and as set of hard constraints H the Boolean axioms. This concludes the proof of Theorem 4.1.

5 Tseitin formulas

We exemplify the saturation process on *Tseitin formulas*, although also Example 3.4 was found using the saturation process. An implementation of the saturation algorithm in python is freely available at `https://github.com/jordilevy/pyPolyCal.git`.

First, we recall what are Tseitin formulas. That is, in this section, consider fixed a graph $G = (V, E)$ with $|V| = 2n + 1$ and Boolean variables $x_{v,w}$ for each $\{v, w\} \in E$. For $v \in V$, let $N(v) = \{w \in V : \{v, w\} \in E\}$. The *Tseitin formula* on G is a CNF formula expressing that in each vertex $v \in V$ the parity of the variables x_e for the edges incident to v is 1, that is Tseitin(G) is the CNF

$$\bigcup_{v\in V} \Big\{ \bigoplus_{w\in N(v)} x_{v,w} = 1 \pmod 2 \Big\}\,, \tag{4}$$

where $\bigoplus_{w\in N(v)} x_{v,w} = 1 \pmod 2$ is encoded as a set of clauses. For instance, if $N(v) = \{w_1, w_2, w_3\}$, then $\bigoplus_{w\in N(v)} x_{v,w} + 1 \pmod 2$ is

$$\{x_{v,w_1}, x_{v,w_2}, x_{v,w_3}\},\ \{\neg x_{v,w_1}, \neg x_{v,w_2}, x_{v,w_3}\},$$
$$\{x_{v,w_1}, \neg x_{v,w_2}, \neg x_{v,w_3}\},\ \{\neg x_{v,w_1}, x_{v,w_2}, \neg x_{v,w_3}\}\ . \tag{5}$$

Since V has an odd size, Tseitin(G) is unsatisfiable.

Consider first the natural encoding of eq. (4) as polynomials. That is consider the set of variables $X = \{x_{v,w} : \{v, w\} \in E\}$ and L_v be the polynomial $\sum_{w\in N(v)} x_{v,w} + 1$ in $\mathbb{F}_2[X]$.

It is well known that $\mathsf{PC}_{\mathbb{F}_2}$ is able to refute $\{L_v : v \in V\}$ in linear size. In $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ we prove more.

▶ **Proposition 5.1.** *There is a linear size derivation in* $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ *of* $[\,1\,,\,c\,]$ *from* $\{[\,L_v\,,\,1\,] : v \in V\}$*, where* c *is the number of connected components of odd size in* G*. In particular, if* G *is connected,* $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ *proves that* $\{[\,L_v\,,\,1\,] : v \in V\}$ *is minimally unsatisfiable, i.e. it is possible to satisfy all polynomials in it except one.*

Proof. We show how to infer $[\,1\,,\,1\,]$ from $\{[\,L_v\,,\,1\,] : v \in V\}$ via the saturation process, when G is connected. For the saturation process, the order in which we saturate the variables is not important.

At each intermediate saturation step ℓ there is a set of weighted polynomials $\mathcal{S}_\ell$ that we have to saturate. The set $\mathcal{S}_\ell$ has the form $\{[\,L_{S_1}\,,\,1\,],\ \ldots, [\,L_{S_m}\,,\,1\,]\}$, where $S_1, \ldots, S_m$ form a partition of V and $L_{S_i} = \sum_{v\in S_i} L_v$. Moreover, we already saturated w.r.t. all the variables $x_{v,w}$ with v, w in the same S_i.

At the beginning of the saturation process, we have the partition of V consisting of all the singletons: $\{\{v\} : v \in V\}$.

Suppose then we are at an intermediate step ℓ of the saturation. We have a set $\mathcal{S}_\ell = \{[\,L_{S_1}\,,\,1\,],\ \ldots, [\,L_{S_m}\,,\,1\,]\}$ and we want to saturate w.r.t. $x_{v,w}$. By the inductive assumption, $\{v, w\}$ is not an internal edge of any of the sets S_is. Hence there are exactly two distinct sets S_i and S_j with $v \in S_i$ and $w \in S_j$. That is, to saturate $\mathcal{S}_\ell$ w.r.t. $x_{v,w}$ is enough to saturate $\mathcal{S}' = \{[\,L_{S_i}\,,\,1\,],\ [\,L_{S_j}\,,\,1\,]\}$. We follow the procedure from Lemma 4.4.

▶ **Fact 1.** *For every linear polynomial* L *depending on* x*,* $L_{x=0}L_{x=1} = L_{x=0}(L_{x=0} + 1) \equiv 0$.

By Fact 1, the only possibility is to SUM L_{S_i} and L_{S_j}. That is from $\mathcal{S}'$ we obtain

$$\mathcal{S}'' = \{[\,L_{S_i} + L_{S_j}\,,\,1\,],\ [\,L_{S_i}L_{S_j}\,,\,2\,]\}\ .$$

Now, $L_{S_i} + L_{S_j} \equiv L_{S_i\cup S_j}$ does not contain variables $x_{v',w'}$ with $v', w' \in S_i \cup S_j$. In particular it does not contain $x_{v,w}$. To continue the saturation process, the only possibility would be to do a SPLIT on $L_{S_i}L_{S_j}$, but this produces the polynomial $L_{S_i,x=0}L_{S_j,x=0}L_{S_i,x=1}L_{S_j,x=1} \equiv 0$ by Fact 1. Therefore $\mathcal{S}''$ is saturated w.r.t. $x_{v,w}$. And so is the multi-set

$$\{[\,L_{S_k}\,,\,1\,] : k \neq i, j\} \cup \{[\,L_{S_i} + L_{S_j}\,,\,1\,],\ [\,L_{S_i}L_{S_j}\,,\,2\,]\}\ .$$

The part of this set not depending on $x_{v,w}$ is

$$\mathcal{S}_{\ell+1} = \{[\,L_{S_k}\,,\,1\,] : k \neq i, j\} \cup \{[\,L_{S_i} + L_{S_j}\,,\,1\,]\}\ .$$

Notice that $[\,L_{S_i}L_{S_j}\,,\,2\,]$ is not in $\mathcal{S}_{\ell+1}$ since it depends on $x_{v,w}$. The set $\mathcal{S}_{\ell+1}$ is the one we want to saturate at the next step for some other variable. During the saturation process we obtain coarser and coarser partitions of V and, at the end of the whole process, we obtain $\{[\,L_V\,,\,1\,]\}$. To conclude we just need to observe that $L_V \equiv |V| \equiv 1$. ◀

To show that $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ and weighted Resolution are incomparable, we need to consider Tseitin(G) encoded as a set of polynomials using the twin variables encoding from Section 2.2. Assume all the initial polynomials of this encoding to have weight 1. From this system of polynomials is still easy to derive $[\,1\,,\,c\,]$ in $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ where c is the number of connected components of odd size in G. Such derivations can be found using the saturation process, provided we use the natural heuristic of preferentially taking the SUM of two weighted polynomials $[\,f\,,\,w\,]$ and $[\,g\,,\,w\,]$ when $fg \equiv 0$. The intuitive reason behind this heuristic is that in such a SUM the number of polynomials in conclusion decreases and we do not introduce a polynomial of higher degree. Under this heuristic it is immediate to see that the saturation process will essentially reconstruct the polynomials $\{[\,L_v\,,\,1\,] : v \in V(G)\}$. Indeed, take for instance the twin variables encoding of the set of clauses in eq. (5), that is

$$S = \{[\,\bar{x}_{v,w_1}\bar{x}_{v,w_2}\bar{x}_{v,w_3}\,,\,1\,],\ [\,x_{v,w_1}x_{v,w_2}\bar{x}_{v,w_3}\,,\,1\,],\\ [\,\bar{x}_{v,w_1}x_{v,w_2}x_{v,w_3}\,,\,1\,],\ [\,x_{v,w_1}\bar{x}_{v,w_2}x_{v,w_3}\,,\,1\,]\}\ .$$

We have that the product of any two of the polynomials is divisible by $x_{v,w_i}\bar{x}_{v,w_i} \equiv 0$ for some i. Therefore applying the SUM rule on S, eventually we will obtain

$$[\,\bar{x}_{v,w_1}\bar{x}_{v,w_2}\bar{x}_{v,w_3} + x_{v,w_1}x_{v,w_2}\bar{x}_{v,w_3} + \bar{x}_{v,w_1}x_{v,w_2}x_{v,w_3} + x_{v,w_1}\bar{x}_{v,w_2}x_{v,w_3}\,,\,1\,] \equiv [\,L_v\,,\,1\,]\ .$$

6 Polynomial Calculus for MaxSAT (general case)

In this section, we adapt the definition of $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ and $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}$ from $\mathbb{F}_2$ to an arbitrary finite field $\mathbb{F}_q$. Polynomial Calculus modulo distinct primes has been studied, for instance in [11]. The Tseitin principle can be extended from counting mod 2 to counting mod p, and this principle is easy in Polynomial Calculus on polynomials with coefficients in $\mathbb{F}_p$.

That is we focus on polynomials with coefficients in $\mathbb{F}_q$, where $q = p^k$ for some fixed prime p and $k \in \mathbb{N}$. Recall that $\mathbb{F}_q$ is the finite field with q elements (this field is unique up to isomorphism), and for each element of $a \in \mathbb{F}$, $a^q = a$ and $p \cdot a = \underbrace{a + \cdots + a}_{p} = 0$.

Fold-unfold and H-equivalence. We consider multi-sets of weighted polynomials, i.e. pairs $[\,f\,,\,w\,]$ with $f \in \mathbb{F}_q[X]$ and $w \in \mathbb{Z}$, under the FOLD-UNFOLD equivalence and the H-EQUIVALENCE. This is the same as what we saw in Section 3 with the only difference that the polynomials instead of being in $\mathbb{F}_2[X]$ now belong to $\mathbb{F}_q[X]$.

Recall that, for f, g H-equivalent we write $f \equiv_H g$. In particular, for every H, $f^q \equiv_H f$. Hence, in the application of the rules, by the H-equivalence we can always assume the variables in all the polynomials to appear with degree at most $q-1$. Moreover, if $H \supseteq \{x^2 - x : x \in X\}$, then we can assume the polynomials to be multilinear. For $H = \{x^2 - x : x \in X\}$, to check efficiently if $f \equiv_H g$ we can then compute the multilinearization of f and g and compare them. In the case of the twin variables $\{x_1, \ldots, x_n, \bar{x}_1, \ldots, \bar{x}_n\}$, $H = \{x_i^2 - x_i, x_i + \bar{x}_i - 1 : i \in [n]\}$, we also have that we can check efficiently if $f \equiv_H g$ using the construction in [19, section 4.3 and Theorem 4.4].

Sum and split. The general forms of the SPLIT and SUM rules are:

$$\frac{[\,f\,,\,w\,]}{[\,fg\,,\,w\,] \quad [\,f(g^{q-1}-1)\,,\,w\,]}\ (\textsc{split})$$

$$\frac{[\,f\,,\,w\,] \quad [\,g\,,\,w\,]}{[\,f+g\,,\,w\,] \quad [\,fg\,,\,w\,] \quad [\,f\,((f+g)^{q-1}-1)\,,\,w\,]}\ (\textsc{sum})$$

for all $f, g \in \mathbb{F}_q[X]$ and $w \in \mathbb{Z}$. The SPLIT rule is sound since for every assignment $\alpha\colon X \to \mathbb{F}_q$ if $f(\alpha) = 0$ the cost of the premise is 0 and so is the cost of the conclusion. If $f(\alpha) \neq 0$, then either $g(\alpha) = 0$ or $g(\alpha) \neq 0$, but in this latter case then $g^{q-1}(\alpha) = 1$. The soundness of the SUM rule is analogous.

Using the rules above, we generalize immediately the definition of $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ and $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}$ (Definition 3.3) from weighted polynomials with coefficients in $\mathbb{F}_2$ to weighted polynomials with coefficients in $\mathbb{F}_q$. We call the resulting systems $\mathsf{wPC}_{\mathbb{F}_q,\mathbb{N}}$ and $\mathsf{wPC}_{\mathbb{F}_q,\mathbb{Z}}$.

Similar to the case of $\mathbb{F}_2$, we have that the SUM rule is redundant in $\mathsf{wPC}_{\mathbb{F}_q,\mathbb{Z}}$. (The proof of this fact will appear in the final version of this paper.)

▶ **Theorem 6.1** (soundness). *Given $F = \{[\, f_1 \,,\, w_1 \,], \ldots, [\, f_m \,,\, w_m \,]\}$ where $f_i \in \mathbb{F}_q[X]$ and a set of polynomials $H \subseteq \mathbb{F}_q[X]$, if there is a $\mathsf{wPC}_{\mathbb{F}_q,\mathbb{Z}}$ derivation of $[\, 1 \,,\, w \,]$ from F (and H as hard constraints), then $\mathrm{cost}_H(F) \geq w$.*

Proof Sketch. This is a simple generalization of the proof of Theorem 3.5. ◀

We show the completeness in the case of Boolean axioms. That is $H = \{x^2 - x : x \in X\}$ or, for the twin variables $\{x_1, \ldots, x_n, \bar{x}_1, \ldots, \bar{x}_n\}$ $H = \{x_i^2 - x_i,\ x_i + \bar{x}_i - 1 : i \in [n]\}$.

▶ **Theorem 6.2** (completeness for Boolean variables). *Given F a multiset of weighted polynomials in $\mathbb{F}_q[X]$, there is a $\mathsf{wPC}_{\mathbb{F}_q,\mathbb{N}}$ derivation of $[\, 1 \,,\, \mathrm{cost}_H(F) \,]$ from F, and the set of Boolean axioms as hard constraints.*

The argument is a minor adaptation of the argument we saw in Section 4. The definition of when a polynomial f depends on a variable x (Definition 4.2), the definition of (x, H)-saturated set (Definition 4.3), and Lemma 4.5 do not really depend on $\mathbb{F}_2$ and can be trivially extended to $\mathbb{F}_q$. Therefore to prove Theorem 6.2 it is enough to show how to adapt Lemma 4.4, the lemma showing how to construct the saturation.

▶ **Lemma 6.3.** *For every set of weighted polynomials S and every variable x, there is a $\mathsf{wPC}_{\mathbb{F}_q,\mathbb{N}}$ derivation of a set of polynomials S' which is (x, H)-saturated, for H the Boolean axioms.*

Proof Sketch. The proof is analogous to the argument for Lemma 4.4. The crucial property used to prove Lemma 4.4 was that $f f_{x=0} f_{x=1} \equiv f_{x=0} f_{x=1}$, which is true for polynomials with coefficients in $\mathbb{F}_2$. For $\mathbb{F}_q$ it is analogous: the crucial property is that $f^{q-1} f_{x=0} f_{x=1} \equiv f_{x=0} f_{x=1}$. This polynomial can be derived from f using the SPLIT rule of $\mathsf{wPC}_{\mathbb{F}_q,\mathbb{N}}$ as follows

$$\dfrac{\dfrac{[\, f \,,\, w \,]}{[\, f f^{q-2} f_{x=0} f_{x=1} \,,\, w \,] \quad [\, f((f^{q-2} f_{x=0} f_{x=1})^{q-1} - 1) \,,\, w \,]}\ \text{SPLIT}}{[\, f_{x=0} f_{x=1} \,,\, w \,] \quad [\, f(f_{x=0} f_{x=1})^{q-1} - f \,,\, w \,]}\ \equiv$$

Notice that this generalizes the case of $\mathbb{F}_2$. Similar to that special case, we have that this SPLIT is non-trivial if both $f_{x=0} f_{x=1} \not\equiv 0$ and $f^{q-2} f_{x=0} f_{x=1} \not\equiv f$, which similarly to the case of $\mathbb{F}_2$ is equivalent to saying that f depends on x. (The proof of this fact will appear in the final version of this paper.) The construction of the sequences of multi-sets is then the natural adaptation of the construction we saw for $\mathbb{F}_2$ to $\mathbb{F}_q$. The reason the sequence is finite and the obtained multi-set is x-saturated is the same as in Lemma 4.4. ◀

7 Conclusions

We showed a way to generalize Polynomial Calculus to the context of MaxSAT, for polynomials with coefficients in a finite field. This involves extending the rules of Polynomial Calculus to have additional conclusions and applying them replacing premises with conclusions, to make them sound for MaxSAT. We showed its completeness via a saturation process. The resulting proof system may be used for SAT or for MaxSAT. The system $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{N}}$ is stronger than MaxSAT Resolution, and $\mathsf{wPC}_{\mathbb{F}_2,\mathbb{Z}}$ is stronger than $\mathbb{Z}$-weighted Resolution (aka Sherali-Adams). As an example, we show how the process is able to prove efficiently Tseitin formulas.

References

1 Carlos Ansótegui, Maria Luisa Bonet, Jordi Levy, and Felip Manyà. The logic behind weighted CSP. In Manuela M. Veloso, editor, *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pages 32–37, 2007. URL: `http://ijcai.org/Proceedings/07/Papers/003.pdf`.

2 Carlos Ansótegui and Jordi Levy. Reducing SAT to Max2SAT. In *Proc. of the 30th International Joint Conference on Artificial Intelligence (IJCAI'21)*, pages 1367–1373, 2021.

3 Albert Atserias, Johannes Klaus Fichte, and Marc Thurley. Clause-learning algorithms with many restarts and bounded-width resolution. In *Lecture Notes in Computer Science*, pages 114–127. Springer Berlin Heidelberg, 2009.

4 Albert Atserias and Massimo Lauria. Circular (yet sound) proofs. In Mikolás Janota and Inês Lynce, editors, *Theory and Applications of Satisfiability Testing - SAT 2019 - 22nd International Conference, SAT 2019, Lisbon, Portugal, July 9-12, 2019, Proceedings*, volume 11628 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2019.

5 Maria Luisa Bonet, Sam Buss, Alexey Ignatiev, Antonio Morgado, and Joao Marques-Silva. Propositional proof systems based on maximum satisfiability. *Artificial Intelligence*, 300:103552, 2021.

6 Maria Luisa Bonet and Jordi Levy. Equivalence between systems stronger than resolution. In Luca Pulina and Martina Seidl, editors, *Theory and Applications of Satisfiability Testing – SAT 2020*, pages 166–181, Cham, 2020. Springer International Publishing.

7 Maria Luisa Bonet, Jordi Levy, and Felip Manyà. A complete calculus for Max-SAT. In *Proc. of the 9th Int. Conf. on Theory and Applications of Satisfiability Testing (SAT'06)*, pages 240–251, 2006.

8 Maria Luisa Bonet, Jordi Levy, and Felip Manyà. Resolution for max-SAT. *Artif. Intell.*, 171(8-9):606–618, 2007.

9 Michael Brickenstein and Alexander Dreyer. Polybori: A framework for Groebner-basis computations with boolean polynomials. *Journal of Symbolic Computation*, 44(9):1326–1345, 2009. Effective Methods in Algebraic Geometry. `doi:10.1016/j.jsc.2008.02.017`.

10 B. Buchberger. A theoretical basis for the reduction of polynomials to canonical forms. *SIGSAM Bull.*, 10(3):19–29, August 1976. `doi:10.1145/1088216.1088219`.

11 Sam Buss, Dima Grigoriev, Russell Impagliazzo, and Toniann Pitassi. Linear gaps between degrees for the polynomial calculus modulo distinct primes. *Journal of Computer and System Sciences*, 62(2):267–289, 2001.

12 Matthew Clegg, Jeff Edmonds, and Russell Impagliazzo. Using the Groebner basis algorithm to find proofs of unsatisfiability. In Gary L. Miller, editor, *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 174–183. ACM, 1996.

13 Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *The Journal of Symbolic Logic*, 44(1):36–50, 1979.

14 Jesús A De Loera, J. Lee, S. Margulies, and S. Onn. Expressing combinatorial problems by systems of polynomial equations and Hilbert's Nullstellensatz. *Comb. Probab. Comput.*, 18(4):551–582, July 2009. `doi:10.1017/S0963548309009894`.

15 Jesús A De Loera, Jon Lee, Peter N Malkin, and Susan Margulies. Computing infeasibility certificates for combinatorial problems through Hilbert's Nullstellensatz. *Journal of Symbolic Computation*, 46(11):1260–1283, 2011.

16 Jesús A De Loera, Susan Margulies, Michael Pernpeintner, Eric Riedl, David Rolnick, Gwen Spencer, Despina Stasi, and Jon Swenson. Graph-coloring ideals: Nullstellensatz certificates, Gröbner bases for chordal graphs, and hardness of Gröbner bases. In *Proceedings of the 2015 ACM on International Symposium on Symbolic and Algebraic Computation*, pages 133–140. ACM, 2015.

17 Susanna F. de Rezende, Massimo Lauria, Jakob Nordström, and Dmitry Sokolov. The Power of Negative Reasoning. In Valentine Kabanets, editor, *36th Computational Complexity Conference (CCC 2021)*, volume 200 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 40:1–40:24, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

18 Jean Charles Faugère. A new efficient algorithm for computing gröbner bases without reduction to zero (f5). In *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation*, ISSAC '02, pages 75–83, New York, NY, USA, 2002. Association for Computing Machinery. doi:10.1145/780506.780516.

19 Yuval Filmus, Edward A. Hirsch, Artur Riazanov, Alexander Smal, and Marc Vinyals. Proving unsatisfiability with hitting formulas. *Electron. Colloquium Comput. Complex.*, TR23-016, 2023. arXiv:TR23-016.

20 Dima Grigoriev. Linear lower bound on degrees of positivstellensatz calculus proofs for the parity. *Theoretical Computer Science*, 259(1-2):613–622, May 2001.

21 Alexey Ignatiev, António Morgado, and João Marques-Silva. On tackling the limits of resolution in SAT solving. In *Proc. of the 20th Int. Conf. on Theory and Applications of Satisfiability Testing (SAT'17)*, pages 164–183, 2017.

22 Daniela Kaufmann, Paul Beame, Armin Biere, and Jakob Nordström. Adding dual variables to algebraic reasoning for gate-level multiplier verification. In *Proceedings of the 25th Design, Automation and Test in Europe Conference (DATE'22)*, 2022.

23 Daniela Kaufmann and Armin Biere. Nullstellensatz-proofs for multiplier verification. In *Computer Algebra in Scientific Computing - 22nd International Workshop, CASC 2020, Linz, Austria, September 14-18, 2020, Proceedings*, pages 368–389, 2020. doi:10.1007/978-3-030-60026-6_21.

24 Daniela Kaufmann, Armin Biere, and Manuel Kauers. Verifying large multipliers by combining SAT and computer algebra. In *2019 Formal Methods in Computer Aided Design, FMCAD 2019, San Jose, CA, USA, October 22-25, 2019*, pages 28–36, 2019. doi:10.23919/FMCAD.2019.8894250.

25 Daniela Kaufmann, Armin Biere, and Manuel Kauers. From DRUP to PAC and back. In *2020 Design, Automation & Test in Europe Conference & Exhibition, DATE 2020, Grenoble, France, March 9-13, 2020*, pages 654–657, 2020. doi:10.23919/DATE48585.2020.9116276.

26 Javier Larrosa and Emma Rollon. Towards a better understanding of (partial weighted) MaxSAT proof systems. In *Proc. of the 23nd Int. Conf. on Theory and Applications of Satisfiability Testing (SAT'20)*, pages 218–232, 2020.

27 Javier Larrosa and Emma Rollón. Augmenting the power of (partial) MaxSAT resolution with extension. In *Proc. of the 34th Nat. Conf. on Artificial Intelligence (AAAI'20)*, 2020.

28 Knot Pipatsrisawat and Adnan Darwiche. On the power of clause-learning SAT solvers as resolution engines. *Artif. Intell.*, 175(2):512–525, 2011.

29 A.A. Razborov. Lower bounds for the polynomial calculus. *Computational Complexity*, 7(4):291–324, December 1998.

30 Emma Rollon and Javier Larrosa. Proof complexity for the maximum satisfiability problem and its use in SAT refutations. *J. Log. Comput.*, 32(7):1401–1435, 2022.

31 Dmitry Sokolov. (Semi)Algebraic proofs over $\{\pm 1\}$ variables. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*. ACM, June 2020.

32 Alasdair Urquhart. Hard examples for resolution. *J. ACM*, 34(1):209–219, January 1987.