Journal of Artificial Intelligence Research 79 (2024) 447-482 Submitted 06/2023; published 02/2024

Weighted, Circular and Semi-Algebraic Proofs

Ilario Bonacina BONACINA@CS.UPC.EDU
Maria Luisa Bonet BONET@CS.UPC.EDU
UPC Universitat Politecnica de Catalunya
Jordi Girona, 1-3
08034 Barcelona, Spain

Jordi Levy LEVY@IIIA.CSIC.ES
Artificial Intelligence Research Institute (IIIA)
Spanish Research Council (CSIC)
Campus UAB Carrer de Can Planas, Zona 2
08193 Barcelona, Spain

Abstract

In recent years there has been an increasing interest in studying proof systems stronger than Resolution, with the aim of building more efficient SAT solvers based on them. In defining these proof systems, we try to find a balance between the power of the proof system (the size of the proofs required to refute a formula) and the difficulty of finding the proofs.

In this paper we consider the proof systems circular Resolution, Sherali-Adams, Nullstellensatz and Weighted Resolution and we study their relative power from a theoretical perspective. We prove that circular Resolution, Sherali-Adams and Weighted Resolution are polynomially equivalent proof systems. We also prove that Nullstellensatz is polynomially equivalent to a restricted version of Weighted Resolution. The equivalences carry on also for versions of the systems where the coefficients/weights are expressed in unary.

The practical interest in these systems comes from the fact that they admit efficient algorithms to find proofs in case these have small width/degree.

1. Introduction

The Satisfiability (SAT) and Maximum Satisfiability (MaxSAT) problems are central in computer science. SAT is the problem of deciding if a given CNF formula has an assignment of 0/1 values that satisfies it. MaxSAT is the optimization version of SAT. Given a CNF formula, we want to know what is the maximum number of clauses that can be satisfied by an assignment. SAT and the decision version of MaxSAT are NP-complete. Problems in many different areas like planning, computational biology, circuit design and verification, etc. can be solved by encoding them into SAT or MaxSAT, and then using a SAT or a MaxSAT solver.

Resolution-based SAT solvers can handle huge industrial instances successfully, but on the other hand, seemingly easy tautologies like the Pigeonhole Principle require exponentially long Resolution proofs (Haken, 1985). An important research direction is to implement SAT-solvers based on stronger proof systems than Resolution. To be able to do that, the proof systems should not be too strong, given that it is commonly believed that the stronger a proof system is, the harder it is to build efficient algorithms to find refutations for the sets of clauses. This is related to the notion of automatability (Bonet, Pitassi, & Raz, 2000; Atserias & Müller, 2019).

To overcome the limitations of Resolution, in the last few years, a number of proof systems somewhat stronger than Resolution or with similar strength have been defined. Among them are proof systems based on MaxSAT, such as MaxSAT Resolution with Extension (Larrosa & Rollón, 2020), or Dual-Rail MaxSAT (Ignatiev et al., 2017, Bonet et al., 2021, Morgado et al., 2019), or Weighted Dual-Rail MaxSAT (Bonet et al., 2018, 2021), or the SAT-to-Max2SAT strategy (Ansótegui & Levy, 2021), or proof systems based on semi-algebraic reasoning, for instance Sherali-Adams (Sherali & Adams, 1994; Dantchev, Martin, & Rhodes, 2009b) and SubCubeSum (Filmus et al., 2020), or proof systems allowing more general proof structures such as circular Resolution (Atserias & Lauria, 2019). A common feature of all these systems is that they have polynomial-size proofs of the Pigeonhole Principle.

Sherali-Adams (SA) is equivalent to *circular Resolution* (c-Res) (Atserias & Lauria, 2019) and low degree version of SA is connected to the TFNP class PPADS (Göös, Hollender, Jain, Maystre, Pires, Robere, & Tao, 2022). MaxSAT Resolution with Extension simulates Dual-Rail MaxSAT (Larrosa & Rollón, 2020), and weighted Dual-Rail MaxSAT simulates Resolution (Bonet et al., 2018).

In the preliminary version of this work, Bonet and Levy (2020) defined *weighted Resolution* (w-Res), a system handling multisets of clauses with integer weights and proved the equivalence between weighted Resolution and circular Resolution. Concurrently and independently, Larrosa and Rollon (2020) and Rollon and Larrosa (2022) also proved a version of this equivalence for a proof system similar to weighted Resolution but working with infinite weights. The definition of weighted Resolution was inspired by the proof system MaxSAT Resolution with Extension (Larrosa & Rollón, 2020). Indeed, weighted Resolution with weights in $\mathbb{Z}$ is equivalent to MaxSAT Resolution with Extension, and it simulates Dual-Rail MaxSAT.

1.1 Our Contributions

This paper uses the framework of weighted Resolution that can be applied to solve both SAT and MaxSAT, the only difference being the presence or not of limitations on the weights of the initial clauses (see also Remark 1.1).

In this paper we substantially extend and simplify the results on weighted Resolution by Bonet and Levy (Bonet & Levy, 2020). In weighted Resolution the initial clauses have positive weights and from them we infer new weighted clauses using substitution rules, i.e. rules that substitute the premises by the conclusions, in the manner of MaxSAT Resolution. Arbitrary clauses may be introduced as assumptions at any time together with the same clause with the corresponding negative weight. The negative weights are a way to control these assumptions. The clauses with negative weights are eliminated by obtaining the same clause with positive weight. At the end of the proof we have a multiset of clauses containing the goal clause. In this multiset all weights must be positive, and therefore all the assumptions will have been justified.

We simplify the equivalence between weighted Resolution and circular Resolution from (Bonet & Levy, 2020) (Theorem 4.3) and we give a direct proof of the equivalence of weighted

Resolution and Sherali-Adams (Theorem 5.7). Unlike (Larrosa & Rollon, 2020; Rollon & Larrosa, 2022) working with finite weights allows us to have an explicit correspondence between weights in weighted proofs and flows in circular proofs.

We also define a restricted version of weighted Resolution where at the end of the proof not only all weights are positive but also all the final clauses, except the goal one, are weakenings of the initial clauses. We prove that restricted weighted Resolution is equivalent to Nullstellensatz (NS), a well-studied algebraic proof system. This is also a new contribution.

Both weighted Resolution and its restricted version might be considered with weights in unary (i.e. only ±1). Those systems are equivalent to unary versions of Sherali-Adams and Nullstellensatz, proof systems that recently have been studied in connection with TFNP complexity classes (Göös et al., 2022). This is also a new contribution.

Fig. 1 summarizes the simulations and equivalences among the proof systems we analyse in this paper.

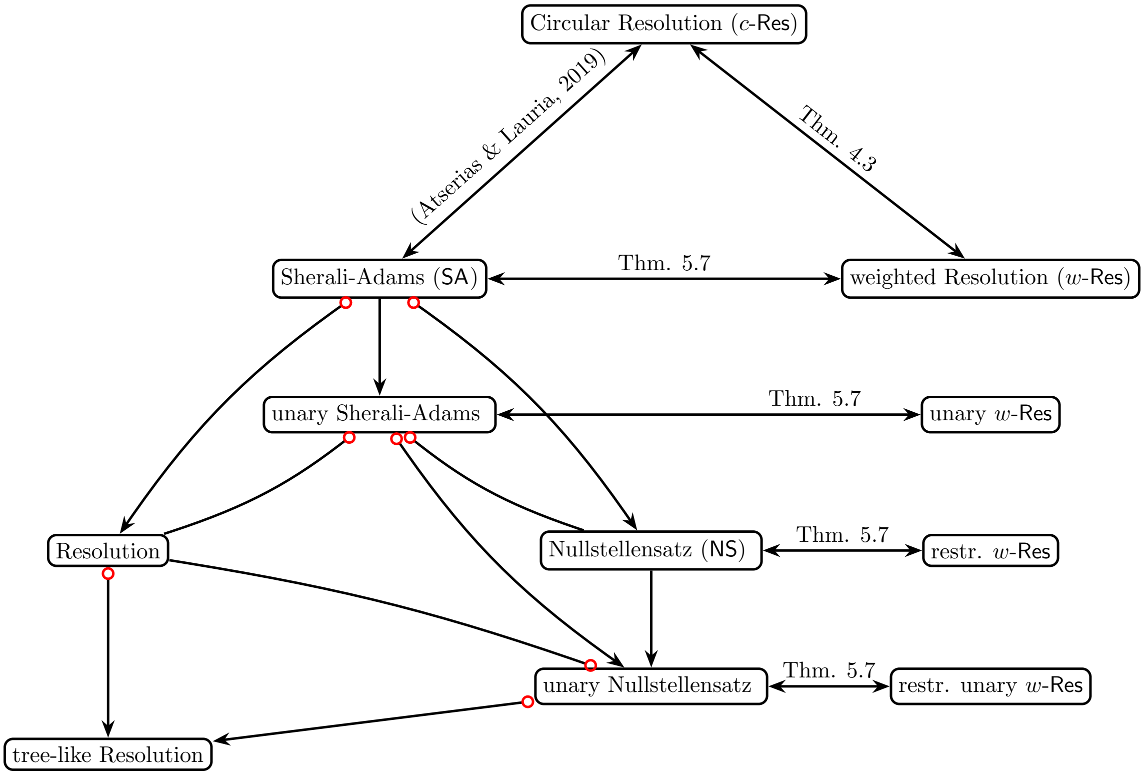


Figure 1: Diagram of known p-simulations. $P \longrightarrow Q$ means that P p-simulates Q. $P \multimap Q$ means that P does not p-simulates Q. We omit from the figure the arrows implied by transitivity. For instance, w-Res $\longrightarrow$ unary w-Res but we omit the arrow.

Remark 1.1 (**SAT vs MaxSAT proofs**). Notice that the format of weighted Resolution can be used both in the context of SAT and weighted MaxSAT, the only difference being that for weighted MaxSAT the weights of the initial clauses are given as part of the input.

Instead, for SAT, the weights of the premises could be arbitrary. The equivalences between the weighted Resolution and the other systems equally hold in the context of MaxSAT.

Bonacina and Bonet (2022) also showed that a well studied logic-based proof system (bounded-depth Frege) plus a weighted version of the Pigeonhole principle (that is given as axioms) simulates Sherali-Adams, and this is a limitation on the strength of the system given that Sherali-Adams efficiently proves that principle. Despite this limitation, we show that Sherali-Adams has polynomial refutations of combinatorial principles like the *HEX principle* (Section see 6), which is shown to be hard for Resolution (Buss, 2006).

1.2 Proof Search Algorithms

One of the reasons to study Sherali-Adams (circular Resolution, weighted Resolution) and Nullstellensatz (restricted weighted Resolution) is that they are in a sweet spot: they are powerful and yet not so powerful as to prohibit the existence of an efficient proof search algorithm (under some restrictions, such as width or degree).

The question of finding refutations is a central question in automated theorem proving and SAT-solving that can be addressed more formally by the notion of *automatability* (Bonet, Pitassi, & Raz, 1997). A proof system P is *automatable* if there is an algorithm that, given as input an unsatisfiable CNF formula F, outputs a refutation of F in P and runs in time polynomial in the shortest P-refutation of F.

Atserias and Müller (2020) showed that Resolution is not automatable, unless $\mathsf{P} = \mathsf{NP}$. Therefore, *any* SAT-solver based on Resolution (for instance CDCL-solvers) will likely require super-polynomial time even on unsatisfiable formulas that are easy to refute in Resolution. Similarly, de Rezende et al. (2021a) showed that NS is not automatable , unless $\mathsf{P} = \mathsf{NP}$. SA is also believed to be not automatable.[1]

Although the above systems are not automatable, they possess a weaker notion of automatability which allows refutations of small width/degree to be found efficiently: the *degree/width-automatability*. Given a formula in n variables, it is well-known that Resolution refutations of width w can be found in time $n^{O(w)}$ (Beame & Pitassi, 1996). Similarly, NS/SA refutations of degree d can be found by linear programming in time $n^{O(d)}$, see for instance the monograph on this subject by Fleming, Kothari, and Pitassi (2019). The fastest known algorithm to search for refutations of a CNF formula F in SA runs in time $2^{O(\sqrt{n \log S})}$, where S is the size of the smallest SA-refutation of F. This follows from Atserias and Hakoniemi's (2019) size-degree tradeoff for SA and the degree-automatability of SA.

Despite the fact that the algorithms to find proofs in SA are not competitive with state-of-the-art SAT-solvers, this view of SA as circular/weighted Resolution might be useful to inspire better algorithms to find refutations in these systems which are stronger than Resolution.

[1] de Rezende et al. (2021a) prove the non-automatability of NS and Polynomial Calculus, a sort of dynamic version of NS. In the same article there is the claim that SA is non-automatable, but this claim has been dropped. The non-automatability of SA is open.

1.3 Organization of the Paper

Section 2 fixes preliminary definitions and notation. In **Section 3**, we introduce the notion of weighted proofs, weighted Resolution and restricted weighted Resolution and we prove basic facts about those systems. In **Section 4**, we recall the notion of circular proofs and circular Resolution and we show the equivalence between weighted and circular Resolution. In **Section 5**, we recall the definitions and basic properties of the algebraic proof systems Nullstellensatz and Sherali-Adams, and then we prove the equivalences between these systems and (restricted) weighted Resolution. In **Section 6**, we give examples of small degree Sherali-Adams refutations of some natural combinatorial principles.

2. Preliminaries

This section contains preliminaries and notation used through the whole paper. We postpone preliminaries only relevant to specific sections. For instance, the notion of *circular proofs* is only used in Section 4 and therefore it is presented there, and similarly the notions of *Nullstellensatz* and *Sherali-Adams* are used in Section 5 and therefore introduced there.

For $n \in \mathbb{N}$, let $[n] := \{1, 2, \ldots, n\}$.

2.1 Clauses and CNF Formulas

A *clause* is a finite disjunction $\vee$ of *literals*, i.e. Boolean variables x or negated Boolean variables $\neg x$. A clause $C = \ell_1 \vee \cdots \vee \ell_r$ has *width* r, noted as $|C| = r$. The empty clause is denoted as $\bot$, and has width 0. A *CNF formula* is a set of clauses. The *size* of a formula is the total number of symbols in it.

A *truth-assignment* is a function $\alpha : \{x_1, \ldots, x_n\} \to \{1, 0\}$, that is a function that assigns to each variable a truth-value *True* (1) or *False* (0). Truth-assignments evaluate literals, clauses, and CNF formulas in the natural way through the usual interpretation of negations and disjunctions. A truth assignment *satisfies* a clause C if it evaluates C to 1, if it evaluates C to 0 we say it *falsifies* F. A clause C is a *logical consequence* of a set of clauses S if every truth-assignment satisfying all clauses in S also satisfies C. For every truth-assignment, the clause $\bot$ evaluates to 0. A truth-assignment *satisfies* a CNF if it satisfies all the clauses in it. A CNF formula is a *contradiction* if every truth-assignment falsifies it.

2.2 Propositional Proof Systems

A *propositional proof system* is a polynomial-time function from $\{0, 1\}^*$ to the set of propositional tautologies. A propositional proof system P *p-simulates* a propositional proof system Q if there is a polynomial-time function f such that $P \circ f = Q$. If P p-simulates Q and Q p-simulates P we say that P and Q are *p-equivalent.*

Several common proof systems are based on inference rules. An *inference rule* is given by two multisets of clauses $\mathcal{F}, \mathcal{G}$. The clauses in $\mathcal{F}$ are called *antecedents* (or premises), the clauses in $\mathcal{G}$ are called *consequents* (or conclusions). We denote inference rules as $\mathcal{F} \vdash \mathcal{G}$ in inline math, and as

$$\frac{\mathcal{F}}{\mathcal{G}}$$

in displayed equations. An *instance* of an inference rule $\mathcal{F} \vdash \mathcal{G}$ is obtained by applying a substitution to the variables of the clauses in $\mathcal{F}$ and $\mathcal{G}$.

Definition 2.1 (soundness)**.** An inference rule $\mathcal{F} \vdash \mathcal{G}$ is *sound* if any truth-assignment that satisfies all clauses in $\mathcal{F}$, also satisfies all the clauses in $\mathcal{G}$.

Definition 2.2 (strong soundness)**.** An inference rule $\mathcal{F} \vdash \mathcal{G}$ is *strongly sound* if for any truth-assignment, the number of falsified clauses in $\mathcal{F}$ equals the number of falsified clauses in $\mathcal{G}$.

The notion of strong soundness was introduced in the context of MaxSAT proof systems as an analogue of the classical soundness.

Definition 2.3 (inference-based proof)**.** Given a set of sound inference rules $\mathcal{R}$, a set $H_1, \ldots, H_m$ of *hypotheses clauses*, and a *goal clause* G, a *proof of* G *from* $H_1, \ldots, H_m$ (using the rules in $\mathcal{R}$) is a sequence of clauses that starts with the clauses $H_1, \ldots, H_m$, ends in G, and such that every clause in the sequence is one of the consequents of an instance of an inference rule from $\mathcal{R}$ with all antecedents appearing earlier in the sequence. A *refutation of* $H_1, \ldots, H_m$ is a proof of $\perp$ from $H_1, \ldots, H_m$. The *size* is the sum of the sizes of all the clauses in the sequence. The *width* is the maximum width of a clause in the sequence.

Resolution The propositional proof system *Resolution* (Res) is a well-known inference-based proof system with inference rules

$$\frac{x \vee A \qquad \neg x \vee B}{A \vee B}\ \textsc{Cut} \qquad \frac{A}{A \vee B}\ \textsc{Weakening} \qquad \frac{}{x \vee \neg x}\ \textsc{Excluded Middle}\quad ,$$

where A, B are clauses and x is a variable. For us, it is convenient to use the split rule instead of the weakening and the symmetric cut instead of the cut:

$$\frac{x \vee A \qquad \neg x \vee A}{A}\ \textsc{Symm. Cut} \qquad \frac{A}{x \vee A \qquad \neg x \vee A}\ \textsc{Split}\quad . \tag{1}$$

This change results in a p-equivalent proof system with rules of Res strongly sound.

Res is sound and complete, that is, if a clause is a logical consequence of a set of clauses there is a Res derivation of it and vice versa.

It is well-known that if a CNF formula in n variables has Resolution refutations of width w then it has a Resolution refutation of size $n^{O(w)}$ (Beame & Pitassi, 1996). This result is tight: Atserias, Lauria, and Nordström (2016) showed that there are 3-CNF formulas that can be refuted in width w but require Resolution size $n^{\Omega(w)}$.

It is also well-known that Res is *width-automatable*, that is, if a CNF formula F in n variables has a Resolution refutation of width w then there is an algorithm running in time $n^{O(w)}$ that finds a Resolution refutation of F. See Section 1.2 for more details on automatability.

2.3 Proof Systems for MaxSAT

Given a CNF formula F as input, SAT is the problem of determining if F is satisfiable, i.e. there exists a truth assignment satisfying all the clauses in F. MaxSAT is the problem of finding an assigment that maximizes the number of satisfied clauses of F (or equivalently, an assignment which minimizes the number of unsatisfied clauses of F).

Inference-based proof systems have also been used in the context of MaxSAT to certify that, for a given CNF formula F, any truth assignment must falsify at least some number k of the clauses in F.

Regarding the inference rules, it is convenient to use strongly sound inference rules (see Definition 2.2) and use them as *substitution* rules.

Definition 2.4 (inference-based proof for MaxSAT)**.** Given a set of strongly sound inference rules $\mathcal{R}$, a multiset of clauses $\{H_1, \ldots, H_m\}$ (*hypotheses*), and a *goal clause* G, a *MaxSAT-proof of* G *from* $H_1, \ldots, H_m$ (using the rules in $\mathcal{R}$) is a sequence of multisets of clauses $\mathcal{L}_1, \ldots, \mathcal{L}_s$ such that

1. $\mathcal{L}_1 = \{H_1, \ldots, H_m\}$,
2. $G \in \mathcal{L}_s$, and
3. for each $i \in [s-1]$, there is a rule in $\mathcal{R}$ of the form $A_1, \ldots, A_\ell \vdash B_1, \ldots, B_r$ such that $\{A_1, \ldots, A_\ell\} \subseteq \mathcal{L}_i$ and $\mathcal{L}_{i+1} = \mathcal{L}_i \setminus \{A_1, \ldots, A_\ell\} \cup \{B_1, \ldots, B_r\}$.

The *size* is the sum of the sizes of all the clauses in the sequence. A proof from $H_1, \ldots, H_m$ of k copies of $\perp$ is certifying that every truth assignment falsifies at least k clauses among $H_1, \ldots, H_m$.

Examples of inference-based proof systems for MaxSAT are MaxSAT-Resolution (Larrosa & Heras, 2005) and the MaxSAT system that uses $\mathcal{R} = \{\text{SPLIT}, \text{SYMM. CUT}\}$. Göös et al. (2022) defined a similar propositional proof system called *reversible* Resolution that also uses the SPLIT and SYMM. CUT rules. The difference between our MaxSAT system and reversible Resolution is that in our system the initial clauses are allowed to appear only once, since we are in the context of MaxSAT.

The notion of an inference-based proof system that replaces premises with conclusions was originally proposed in the context of MaxSAT. However it can also be used in the context of SAT, since a derivation of a $\perp$ certifies the unsatifiability of the initial CNF formula.

In the context of showing unsatisfiability of a set of clauses, these two systems above are p-simulated by Res, and probably strictly weaker, given that the rules are substitution rules and the same clause cannot be used multiple times. Some progress towards solving this open problem was achieved by Py et al. (2022) and Cherif et al. (2022). They show that if the Resolution refutation is “crossing-free”, then it can be converted into MaxSAT Resolution with a polynomial increase in size.

There are several possible ways to overcome the limitations imposed by the use of substitution rules. The easiest one is to allow an arbitrary number of multiple copies of the initial clauses or to use initial clauses with weights. Another is to use clever encodings of the initial clauses and then use the MAXSAT-RESOLUTION rule. Examples of these encodings are the Dual-Rail encoding of Ignatiev et al. (2017), Morgado et al. (2019), and Bonet et al. (2018, 2021), and the SAT-to-Max2SAT strategy of Ansótegui and Levy (2021).

The use of weighted clauses can be extended to a more powerful system allowing negative weights, intuitively meaning that a negative-weighted clause is an assumption yet to be justified.

3. Weighted Proofs and Weighted Resolution

In this section we develop formally the notion of *weighted* clauses and *weighted* Resolution with weights in $\mathbb{Z}$.

Definition 3.1 (weighted clause)**.** A *weighted clause* is a pair (C, w) where C is a clause and $w \in \mathbb{Z}$. The *width* of a weighted clause (C, w) is the width of the clause C.

We consider multisets of weighted clauses modulo an equivalence relation, the *fold-unfold equivalence.*

Definition 3.2 (fold-unfold equivalence)**.** Two multisets of weighted clauses $\mathcal{F}$ and $\mathcal{G}$ are *fold-unfold equivalent* ($\mathcal{F} \approx \mathcal{G}$) if for every clause C,

$$\sum_{u:(C,u)\in\mathcal{F}} u = \sum_{u:(C,u)\in\mathcal{G}} u \ .$$

For instance, for any clause C and $u \in \mathbb{Z}$,

- $\{(C, 2u)\} \approx \{(C, u), (C, u)\}$,
- $\emptyset \approx \{(C, 0)\}$,
- $\emptyset \approx \{(C, u), (C, -u)\}$.

The fold-unfold equivalence between two multisets of weighted clauses may be seen as the application of the following inference rules as substitution rules:

$$\frac{(C,u) \qquad (C,v)}{(C,u+v)}\ \textsc{Fold}\ , \qquad \frac{(C,u+v)}{(C,u) \qquad (C,v)}\ \textsc{Unfold}\ ,$$

$$\frac{}{(C,0)}\ \text{0-}\textsc{Fold}\ , \qquad \frac{(C,0)}{}\ \text{0-}\textsc{Unfold}\ .$$

where C is a clause and $u, v \in \mathbb{Z}$.

The rules Fold/Unfold were previously considered in the context of weights in $\mathbb{N}$, for instance in (Bonet et al., 2021) under the name Extraction/Contraction. The use of negative weights was used for the first time in the context of Resolution in (Larrosa & Rollón, 2020). In the context of Constraint Satisfaction Problems, negative weights were introduced by Cooper, de Givry, and Schiex (2007).

The inference rules generalize to weighted clauses and the same happens with the notion of strongly sound inference rule (see Definition 2.2). Given any inference rule $A_1, \ldots, A_s \vdash B_1, \ldots, B_r$ we can transform it into the inference rule $(A_1, u), \ldots, (A_s, u) \vdash (B_1, u), \ldots, (B_r, u)$ on weighted clauses, where $u \in \mathbb{Z}$.

The following definition is the generalization of the notion of strong soundness to the context of weighted clauses.

Definition 3.3 (soundess for weighted inferences)**.** The inference rule $\mathcal{F} \vdash \mathcal{G}$ on weighted clauses is *(strongly) sound* if for any truth-assignment, the total weight of the falsified clauses in $\mathcal{F}$ equals the total weight of the falsified clauses in $\mathcal{G}$, i.e. for every truth-assignment α,

$$\sum_{\substack{(C,u)\in\mathcal{F} \\ \alpha(C)=0}} u = \sum_{\substack{(C,u)\in\mathcal{G} \\ \alpha(C)=0}} u \ .$$

We give now a notion of inference-based proof system in this context of weighted clauses. We use the fold-unfold equivalence and the inference rules as substitution rules, but we have to be careful with the *negative* weights. For instance, we saw that $\emptyset \approx \{(C, u), (C, -u)\}$, this does not mean we have "derived" the weighted clause (C, u), but, we could see this as the fact that we are assuming the weighted clause (C, u), and $(C, -u)$ is there as "bookkeeping": a reminder that eventually we have to prove (C, u).

Formally the definition is the following.

Definition 3.4 (weighted proofs)**.** Given a set of strongly sound inference rules $\mathcal{R}$ for weighted clauses, a set $\mathcal{H}$ of weighted clauses as *hypotheses*, and a goal weighted clause (C, u) with $u > 0$, a *weighted proof of* (C, u) *from* $\mathcal{H}$ is a sequence of multisets of weighted clauses $(\mathcal{F}_1, \ldots, \mathcal{F}_s)$ such that:

1. $\mathcal{F}_1 = \mathcal{H}$.

2. For every proof step $i \in [s-1]$, either it is
   a **fold-unfold equivalence**: $\mathcal{F}_{i+1} \approx \mathcal{F}_i$,
   or a **regular inference**: an instance of a rule in $\mathcal{R}$ of the form

$$\frac{(A_1,\ u_1) \cdots (A_s,\ u_s)}{(B_1,\ w_1) \cdots (B_r,\ w_r)}$$

   where $\mathcal{F}_{i+1} = \mathcal{F}_i \cup \{(A_1, -u_1), \ldots, (A_s, -u_s),\ (B_1, w_1), \ldots, (B_r, w_r)\}$.

3. $\mathcal{F}_s$ contains (C, u), and possibly other weighted clauses, all of them with *non-negative* weights.

The *size* of a weighted proof is the total number of symbols occurring in $\mathcal{F}_1, \ldots, \mathcal{F}_s$ including the encoding of the weights. A weighted *refutation* over $\mathcal{R}$ is a weighted proof of $(\bot, u)$ for some positive u.

Notice that, in presence of fold-unfold equivalence item 2 in the definition of weighted proofs is equivalent to

2′. For every proof step i, either it is
   a **fold-unfold equivalence**: $\mathcal{F}_{i+1} \approx \mathcal{F}_i$,
   or a **regular inference**: an instance of a rule in $\mathcal{R}$ of the form

$$\frac{(A_1,\ u_1) \cdots (A_s,\ u_s)}{(B_1,\ w_1) \cdots (B_r,\ w_r)}$$

   such that $\{(A_1, u_1), \ldots, (A_s, u_s)\} \subseteq \mathcal{F}_i$ and
   $\mathcal{F}_{i+1} = \mathcal{F}_i \setminus \{(A_1, u_1), \ldots, (A_s, u_s)\} \cup \{(B_1, w_1), \ldots, (B_r, w_r)\}$.

We use this alternative presentation in the examples since it allows the use of slightly smaller multisets of clauses. In the proofs we instead use the presentation from the definition.

A special case of weighted proofs is *weighted Resolution.*

Definition 3.5 (weighted Resolution, w-Res)**.** A *weighted Resolution* (w-Res) derivation is a weighted proof where the set of inference rules for weighted clauses are the following, used as substitution rules:

$$\frac{(x \lor A, u) \qquad (\neg x \lor A, u)}{(A, u)}\ \textsc{Symmetric-Cut} \qquad\qquad \frac{(A, u)}{(x \lor A, u) \qquad (\neg x \lor A, u)}\ \textsc{Split}$$

$$\frac{}{(x \lor \neg x, u)}\ \textsc{Excluded Middle}$$

where A is a clause, x is a variable, and $u \in \mathbb{Z}$. Notice that, due to the fold-unfold, the Split and Symmetric-Cut can be seen as the same rule with opposite weights (one positive and one negative).

Without loss of generality we assume the weights of the application of the Excluded Middle to be always positive.

The definition of w-Res generalizes ideas of Larrosa and Heras (2005) and Bonet et al. (2006, 2007) for weights in $\mathbb{N}$ and was introduced by Larrosa and Rollón (2020). The notion of w-Res could be generalized further to clauses with weights in arbitrary ordered monoids, like $\mathbb{Q}$ or $\mathbb{Z} \cup \{\infty\}$. For simplicity, we focus on weights in $\mathbb{Z}$ since using weights in $\mathbb{Q}$ gives a p-equivalent system. This is easy to see by multiplying all the weights in $\mathbb{Q}$ by the minimum common multiple of the denominators.

Notice that, we have two natural ways of representing the weights. We can represent a weight u in *unary*, or we can represent u in *binary* using $(\lceil \log |u| \rceil + 1)$-many bits. By the fold-unfold equivalence, the unary representation can be equivalently seen as restricting all the weights appearing in a weighted proof to be ± 1.

Another way of restricting w-Res is by restricting the condition on the final multiset of a w-Res derivation.

Definition 3.6 (restricted weighted proofs, restricted w-Res)**.** Given a set of strongly sound inference rules $\mathcal{R}$ for weighted clauses, a set $\mathcal{H}$ of weighted clauses as *hypotheses*, and a goal weighted clause (C, u) with $u > 0$, a *restricted* weighted proof of (C, u) from $\mathcal{H}$ is a sequence of multisets of weighted clauses $(\mathcal{F}_1, \ldots, \mathcal{F}_s)$ satisfying the same conditions of Definition 3.4 with the exception of item 3. Instead, the multiset $\mathcal{F}_s$ satisfies the following condition.

3′. $\mathcal{F}_s$ contains (C, u), and possibly other weighted clauses, all of them with non-negative weights and weakening of weighted clauses in $\mathcal{H}$, i.e. of the form $(C \lor D, w)$ where $(C, w') \in \mathcal{H}$, D is some clause, and $w, w' \in \mathbb{Z}$.

If we consider this for the set of inference rules in Definition 3.5, we call it *restricted* w-Res.

The system restricted w-Res is similar to the system *reversible Resolution with terminals* defined in (Göös et al., 2022). The difference between our system and theirs is that in reversible Resolution with terminals the weights are only allowed to be 1, therefore the clauses are considered with multiplicity.

(Restricted) weighted proofs make sense both in the context of SAT and MaxSAT. If we have a CNF formula $F = \{C_1, \ldots, C_m\}$ over n variables we may work using some set of inference rules and use a (restricted) weighted proof to show the unsatisfiability of F: we have the freedom to choose the weights to give to $C_1, \dots, C_m$, i.e. we look for weighted refutations of $\{(C_1, u_1), \dots, (C_m, u_m)\}$ for some $u_i \in \mathbb{N}$. In the case of MaxSAT we don't have this freedom: all the u_is are 1. Similarly, in weighted MaxSAT the u_is are also given.

Theorem 3.7 (Soundness of weighted proofs)**.** *Let $\mathcal{R}$ a set of strongly sound inference rules on weighted clauses and $\mathcal{F}$ a multiset of weighted clauses. If there exists a weighted proof of $(\bot, s)$ from $\mathcal{F}$ (using the rules in $\mathcal{R}$), then every truth-assignment must falsify clauses in $\mathcal{F}$ with combined weight at least s, i.e. for every truth-assignment α*

$$\sum_{\substack{(C,u)\in\mathcal{F}\\ \alpha(C)=0}} u \geq s \ .$$

Proof. Let $(\mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_\ell)$ be a weighted proof of $(\bot, s)$ from $\mathcal{F}$. Let α be a truth-assignment and let

$$\mathrm{cost}(\mathcal{F}_i, \alpha) = \sum_{\substack{(C,u)\in\mathcal{F}_i\\ \alpha(C)=0}} u \ ,$$

that is, $\mathrm{cost}(\mathcal{F}_i, \alpha)$ is the total-weight of the clauses from $\mathcal{F}_i$ falsified by α.

Since $\mathcal{F} = \mathcal{F}_1$, our goal is to show that $\mathrm{cost}(\mathcal{F}_1, \alpha) \geq s$. By the soundness of the weighted inference rules, we have that

$$\mathrm{cost}(\mathcal{F}_1, \alpha) = \mathrm{cost}(\mathcal{F}_2, \alpha) = \cdots = \mathrm{cost}(\mathcal{F}_\ell, \alpha) \ ,$$

and $\mathrm{cost}(\mathcal{F}_\ell, \alpha) = s + u_1 + \cdots + u_r$ where $u_1, \dots, u_r$ are the weights of the other clauses in $\mathcal{F}_\ell$ falsified by α. All the weighted clauses in $\mathcal{F}_\ell$ must have positive weights, hence, $\mathrm{cost}(\mathcal{F}_\ell, \alpha) \geq s$ and therefore $\mathrm{cost}(\mathcal{F}_1, \alpha) \geq s$. □

Notice that this proof generalizes trivially to sets of rules $\mathcal{R}$ satisfying a relaxed version of the strong soundness, where whenever $\mathcal{F} \vdash \mathcal{G}$ the total weight of the falsified clauses in $\mathcal{F}$ is *at least* the total weight of the falsified clauses in $\mathcal{G}$. That is for every truth-assignment α,

$$\sum_{\substack{(C,u)\in\mathcal{F}\\ \alpha(C)=0}} u \geq \sum_{\substack{(C,u)\in\mathcal{G}\\ \alpha(C)=0}} u \ .$$

Corollary 3.8 (soundness of (restricted) w-Res)**.** *Both w-*Res *and restricted w-*Res *are sound.*

Theorem 3.9 (completeness of (restricted) w-Res)**.** *Both w-*Res *and restricted w-*Res *are complete, i.e. given a multiset of weighted clauses $\mathcal{F}$, if for every truth-assignment α*

$$\sum_{\substack{(C,u)\in\mathcal{F}\\ \alpha(C)=0}} u \geq s \ ,$$

*then there exists a (restricted) w-*Res *derivation of $(\bot, s)$ from $\mathcal{F}$.*

Proof. Using the fold-unfold equivalence expand each clause (C, u) in $\mathcal{F}$ into u copies of $(C, 1)$. Then apply the split to each of those clauses on all the possible variables. If the clause C had width w all the split rules generate 2^{n-w} many clauses of width n and all of them weakening of C. After having done this for each clause in $\mathcal{F}$, let $\mathcal{L}$ be the obtained multiset.

Let $\mathtt{CT}_n$ be the *complete tree* CNF formula on n variables, i.e. the set containing all possible 2^n distinct clauses of width exactly n. Let $w\mathtt{CT}_n = \{(C, 1) \ : \ C \in \mathtt{CT}_n\}$, be the weighted version of $\mathtt{CT}_n$. The multiset $\mathcal{L}$ contains s copies of $w\mathtt{CT}_\mathtt{n}$, where s is the minimum weight of the falsified clauses in $\mathcal{F}$. To see this, suppose in $\mathcal{L}$ there are k copies of $w\mathtt{CT}_\mathtt{n}$ together with a satisfiable set of clauses A. Then there is a total assignment that satisfies all the clauses in A and falsifies exactly one clause in each $w\mathtt{CT}_\mathtt{n}$, for a total of k clauses. By the strong-soundness then $k \geq s$.

It is immediate to see that $w\mathtt{CT}_n$ has a w-Res derivation of $(\bot, 1)$ where one only uses the symm. cut rule and the last multiset is just $\{(\bot, 1)\}$ (using Definition 3.4 item $2'$). Therefore, form $\mathcal{L}$ just using the symm. cut rule we obtain a $(\bot, s)$. Since we apply the split rule only to weakenings of the initial axioms, the w-Res derivation we just described is also a restricted w-Res derivation. □

Notice that, the completeness of weighted MaxSAT Resolution, proved by Bonet et al. (2007), already implies the completeness of w-Res, since w-Res p-simulates weighted MaxSAT Resolution.

We conclude this section showing a sort of normal form for (restricted) w-Res derivations.

Lemma 3.10 (Normal form of w-Res)**.** *Any w-Res derivation $(\mathcal{F}_1, \dots, \mathcal{F}_s)$ can be converted in polynomial time, with a polynomial increase in size and identical width, to a w-Res derivation $(\mathcal{F}'_1, \dots, \mathcal{F}'_s)$ where all the regular inference steps have positive weights and there is a single application of the fold-unfold rule between $\mathcal{F}'_{s-1}$ and $\mathcal{F}'_s$. The same holds for restricted w-Res too.*

Proof. Consider the w-Res derivation $(\mathcal{F}_1, \dots, \mathcal{F}_s)$. First we observe how to get rid of the negative weights in the applications of inference rules which could only be in Split and Symm. Cut.

If from $\mathcal{F}_i$ to $\mathcal{F}_{i+1}$ we applied a symm. cut $\{(x \vee A, -u),\ (\neg x \vee A, -u)\} \vdash (A, -u)$ with $u \geq 0$ we have

$$\mathcal{F}_{i+1} = \mathcal{F}_i \cup \{(x \vee A, u), (\neg x \vee A, u), (A, -u)\} \ ,$$

which is the same as applying the split rule $(A, u) \vdash \{(x \vee A, u),\ (\neg x \vee A, u)\}$ on $\mathcal{F}_i$. Similarly, for the split $(A, -u) \vdash \{(x \vee A, -u),\ (\neg x \vee A, -u)\}$ with $u \geq 0$ we use the symm. cut rule with weight u.

Now, we want to push all the fold-unfold towards the end of the derivation. Let R be an instantiation of a rule and S_R be the set of weighted clauses corresponding to the premises and conclusions of the rule, where the premises will have negative weight and the conclusions will have positive weight. We have that

$$\dfrac{\dfrac{\mathcal{F} \cup \{(C, u_1) \dots (C, u_s)\}}{\mathcal{F} \cup \{(C, v_1) \dots (C, v_r)\}} \approx (u_1 + \ldots + u_s = v_1 + \ldots + v_r)}{\mathcal{F} \cup \{(C, v_1) \dots (C, v_r)\} \cup S_R} \ \mathrm{R}$$

and

$$\dfrac{\dfrac{\mathcal{F} \cup \{(C, u_1) \dots (C, u_s)\}}{\mathcal{F} \cup \{(C, u_1) \dots (C, u_s)\} \cup S_R} \ \mathrm{R}}{\mathcal{F} \cup \{(C, v_1) \dots (C, v_r)\} \cup S_R} \approx (u_1 + \ldots + u_s = v_1 + \ldots + v_r)$$

are both valid derivations and the increase in size is just $|S_R|$. This means that with just a polynomial increase in size we can push all the $\approx$ at the end of the derivation. Now, by the transitivity of $\approx$, all of them can be done at once using fold-unfold equivalence. □

4. Circular Proofs as Weighted Proofs

We show that weighted proofs are the equivalent to *circular proofs*. Atserias and Lauria (2019) introduced the notion of *circular proofs* as a way to enrich the structure of the proofs allowed in Frege systems. We recall the notion of circular Resolution and then we show the equivalence with weighted Resolution.

Definition 4.1 (circular Proof)**.** Given a set of inference rules $\mathcal{R}$, a set of hypotheses clauses $\mathcal{H}$ and a goal clause G, a *circular derivation of G from $\mathcal{H}$ (over $\mathcal{R}$)* is a bipartite directed graph $(\mathcal{I}, \mathcal{J}, E)$ where $\mathcal{I}, \mathcal{J}$ are multisets, nodes are either inference rules ($R \in \mathcal{I}$) or clauses ($A \in \mathcal{J}$), edges $A \to R \in E$ denote the occurrence of clause A as an antecedent of the rule R, and edges $R \to A \in E$ the occurrence of clause A as a consequence of the rule R.

A function $\mathsf{Flow} : \mathcal{I} \to \mathbb{N}^+$ is called *flow-assignment*. Given a flow-assignment Flow, the *balance* of a clause A is

$$\mathrm{Bal}(A, \mathsf{Flow}) = \sum_{R \in N^{\mathrm{in}}(A)} \mathsf{Flow}(R) - \sum_{R \in N^{\mathrm{out}}(A)} \mathsf{Flow}(R) \ ,$$

where $N^{\mathrm{in}}(A) = \{R \in \mathcal{I} \mid R \to A \in E\}$ and $N^{\mathrm{out}}(A) = \{R \in \mathcal{I} \mid A \to R \in E\}$ are the sets of neighbours of A. We will use $\mathrm{Bal}(A)$ to mean $\mathrm{Bal}(A, \mathsf{Flow})$ when the Flow function is clear from the context.

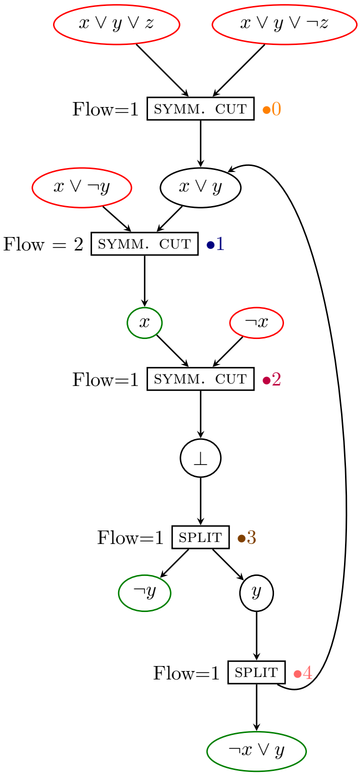
To ensure the soundness, we require the existence of a *flow-assignment* Flow satisfying $\mathrm{Bal}(A, \mathsf{Flow}) \geq 0$, for any clause $A \in \mathcal{J} \setminus \mathcal{H}$, and $\mathrm{Bal}(G, \mathsf{Flow}) > 0$, for the goal clause G. The clauses in $\mathcal{H}$ may have negative balance. The *size* of a circular proof derivation is the number of nodes in the bipartite graph and the *width* is the maximum width of a clause in it.

Definition 4.2 (circular Resolution, c-Res)**.** A *circular Resolution* (c-Res) derivation is a circular proof using the set of inference rules $\mathcal{R} = \{$split, symmetric cut$\}$.

Theorem 4.3. *c-Res and w-Res are p-equivalent. Moreover, the width is either preserved (w-Res p-simulation of c-Res) or it increases by one (c-Res p-simulation of w-Res).*

The two simulations in this theorem are proved separately in Lemma 4.5 and Lemma 4.6 below. Before proving formally the connections between c-Res derivations and w-Res derivations we give an example.

Example 4.4. We give an example of a c-Res derivation of $\{x, \neg y, \neg x \lor y\}$ from $\{x \lor y \lor z, x \lor y \lor \neg z, x \lor \neg y, \neg x\}$. See Figure 2 and Figure 4. Next to the c-Res derivation we show the corresponding w-Res derivation. To improve readability we apply immediately after each inference rule a fold-unfold equivalence. In the c-Res derivation we highlight in red the premises and in green the conclusions. Notice that in the SYMM. CUT •1, we need to apply it with both premises with weight 2, and since $x \lor y$ only has weight 1, we are left with $x \lor y$ with weight -1 in the conclusions of the rule, and therefore it needs to be justified. Notice also that we proved the clause x with weight 2, but it is only used as a premise with weight 1, hence it remains in the conclusions of the proof.



$$\begin{array}{c l}
(x \lor y \lor z, 1)\ (x \lor y \lor \neg z, 1) & \\
(x \lor \neg y, 2)\ (\neg x, 1) & \\ \hline
 & \text{SYMM.CUT}\ \& \approx \bullet 0 \\
(x \lor \neg y, 2)\ (\neg x, 1)\ (x \lor y, 1) & \\ \hline
 & \text{SYMM.CUT}\ \& \approx \bullet 1 \\
(\neg x, 1)\ (x, 2)\ (x \lor y, -1) & \\ \hline
 & \text{SYMM.CUT}\ \& \approx \bullet 2 \\
(\bot, 1)\ (x, 1)\ (x \lor y, -1) & \\ \hline
 & \text{SPLIT}\ \& \approx \bullet 3 \\
(y, 1)\ (\neg y, 1)\ (x, 1)\ (x \lor y, -1) & \\ \hline
 & \text{SPLIT}\ \& \approx \bullet 4 \\
(\neg y, 1)\ (\neg x \lor y, 1)\ (x, 1) &
\end{array}$$

Figure 2: A circular proof and its corresponding weighted proof.

Lemma 4.5. *c-Res p-simulates w-Res. The weights of w-Res become the flows of c-Res (except on the initial clauses where they take the opposite sign).*

Proof. Assume we have a w-Res derivation $(\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_\ell)$ of a weighted clause (C, w) with $w > 0$, from a multiset of weighted clauses $\mathcal{F}$. That is, in particular, $\mathcal{F}_1 = \mathcal{F}$ and $\mathcal{F}_\ell$

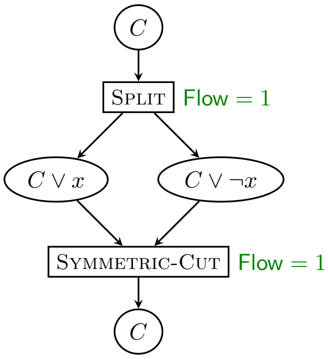


Figure 3: A constant size c-Res derivation of C from C.

contains the weighted clause (C, w) and the rest of weighted clauses in $\mathcal{F}_\ell$ have positive weights. Let $\mathcal{H}$ be the set of distinct clauses appearing in $\mathcal{F}$.

If the goal clause C is already in $\mathcal{H}$, we just construct the c-Res derivation of C from C appearing in Figure 3. This c-Res derivation might increase the width in 1 and in a c-Res derivation the goal clause needs to have positive balance. Therefore the multiset of inference nodes cannot be empty since then C would have balance 0.

In the rest of the argument we assume the goal clause $C \notin \mathcal{H}$. By Lemma 3.10, we can suppose in $(\mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_\ell)$ the regular inference rules are always applied with positive weights and there is a single fold-unfold step between $\mathcal{F}_{\ell-1}$ and $\mathcal{F}_\ell$.

Recall that, if from $\mathcal{F}_i$ to $\mathcal{F}_{i+1}$ we apply the rule

$$\frac{(A_1,\ u) \cdots (A_s,\ u)}{(B_1,\ u), \dots, (B_r,\ u)}\ \text{Rule R} \tag{2}$$

where $u > 0$, then $\mathcal{F}_{i+1} = \mathcal{F}_i \cup \{(A_1,\ -u), \dots, (A_s,\ -u),\ (B_1,\ u) \cdots (B_r,\ u)\}$.

We construct now a c-Res derivation associated to $(\mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_\ell)$ and at the same time a flow-assignment. Consider the *set* J of all the distinct (unweighted) clauses appearing in $\bigcup_{i=1}^{l} \mathcal{F}_i$ and the *multiset* I of all the regular inference rules in $(\mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_\ell)$, i.e. we do not consider the last fold-unfold inference step.

For each instantiation of a Rule R as in eq. (2), we have an inference node R, we set $\mathsf{Flow}(\mathrm{R}) = u$ and we create edges $A_i \to \mathrm{R}$, for $1 = 1, \dots, s$, and $\mathrm{R} \to B_j$, for $j = 1, \dots, r$. Let G be the directed graph resulting from this construction. For each clause node A in G we have that its balance is

$$\mathrm{Bal}(A, \mathsf{Flow}) = \sum_{R \in N^{\mathrm{in}}(A)} \mathsf{Flow}(\mathrm{R}) - \sum_{R \in N^{\mathrm{out}}(A)} \mathsf{Flow}(\mathrm{R}) \ .$$

From the construction of G and the assumptions on $(\mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_\ell)$, for each clause $A \notin \mathcal{H}$ we have

$$\{(A, \mathsf{Flow}(\mathrm{R})) : \mathrm{R} \in N^{\mathrm{in}}(A)\} \cup \{(A, -\mathsf{Flow}(\mathrm{R})) : \mathrm{R} \in N^{\mathrm{out}}(A)\} \subseteq \mathcal{F}_{\ell-1} \ ,$$

that is, in $\mathcal{F}_\ell$, we have the clause $(A, \mathrm{Bal}(A))$. By assumption this is a w-Res derivation of (C, w), therefore it must be that $\mathrm{Bal}(A) \geq 0$ for each $J \setminus \mathcal{H}$ and $\mathrm{Bal}(C) > 0$. □

Lemma 4.6. *w-Res p-simulates c-Res. Given a flow-assignment of a c-Res proof, the corresponding balance will be the weights in w-Res (exept for the initial clauses where it takes the opposite sign).*

Proof. Assume we have a c-Res derivation (I, J, E) with clauses nodes J, inference nodes I, edges E, hypotheses $\mathcal{H} \subset J$ and goal clause $C \in J$. That is, there is a Flow assignment such that for each $A \in J$

$$\mathrm{Bal}(A, \mathsf{Flow}) = \sum_{R \in N^{\mathrm{in}}(A)} \mathsf{Flow}(\mathrm{R}) - \sum_{R \in N^{\mathrm{out}}(A)} \mathsf{Flow}(\mathrm{R}) \ , \tag{3}$$

where $\mathrm{Bal}(C, \mathsf{Flow}) > 0$, and for each $A \in J \setminus \mathcal{H}$, $\mathrm{Bal}(A, \mathsf{Flow}) \geq 0$. Moreover, by elementary facts of linear programming, we can assume that for each $\mathrm{R} \in I$, $\mathsf{Flow}(\mathrm{R})$ has bit-complexity polynomial in $|I| + |J|$.

Without loss of generality, we assume that the hypotheses clauses do not have incoming edges, i.e. for any $A \in \mathcal{H}$, $N^{\mathrm{in}}(A) = \emptyset$. If there was an incoming edge to $A \in \mathcal{H}$, we can make a copy of A and use it to redirect the previous edge from the old $A \in \mathcal{H}$ to the new copy of A. Notice that redirecting these incoming edges in a circular proof only decreases the balance of hypotheses clauses (that are already allowed to have negative balance) and increases the balance of the copy. Without loss of generality we also assume the goal clause C is not in $\mathcal{H}$.

We produce a w-Res derivation by first placing the hypotheses $\mathcal{H}$ at the beginning, then introducing all the remaining clauses $C \in J \setminus \mathcal{H}$, and, finally, applying all the rules $R_1, \ldots, R_{|I|}$ one by one with appropriate weights.

More precisely, we construct multisets $\mathcal{F}_0'', \mathcal{F}_0', \mathcal{F}_0, \mathcal{F}_1, \ldots, \mathcal{F}_{|I|}, \mathcal{F}_{|I|+1}$ such that in this order they form a valid w-Res derivation of $(C, \mathrm{Bal}(C, \mathsf{Flow}))$.

The first multiset $\mathcal{F}_0''$ corresponds to all the hypotheses clauses with the appropriate weight:

$$\mathcal{F}_0'' = \{(A, -\mathrm{Bal}(A)) \mid A \in \mathcal{H}\} \ .$$

The second multiset $\mathcal{F}_0'$ is obtained from $\mathcal{F}_0''$ by fold-unfold equivalence and is the following:

$$\begin{aligned} \mathcal{F}_0' = \ & \{(A, -\mathrm{Bal}(A)) & \mid A \in \mathcal{H}\} \cup \\ & \{(A, \mathrm{Bal}(A)),\ (A, -\mathrm{Bal}(A)) & \mid A \in J \setminus \mathcal{H}\} \ . \end{aligned}$$

The third multiset $\mathcal{F}_0$ is again fold-unfold equivalent to $\mathcal{F}_0'$ and is obtained expanding $-\mathrm{Bal}(A)$ for all $A \in J$:

$$\begin{aligned} \mathcal{F}_0 = \ & \{(A, \mathrm{Bal}(A)) & \mid A \in J \setminus \mathcal{H}\} \cup \\ & \{(A, \mathsf{Flow}(R)) & \mid A \in J \wedge R \in N^{\mathrm{out}}(A)\} \cup \\ & \{(A, -\mathsf{Flow}(R)) & \mid A \in J \wedge R \in N^{\mathrm{in}}(A)\} \ . \end{aligned}$$

For $i = 1, \ldots, |I|$ we define

$$\begin{aligned} \mathcal{F}_i = \ & \mathcal{F}_{i-1} \cup \\ & \{(A, -\mathsf{Flow}(R_i)) & \mid A \in N^{\mathrm{in}}(R_i)\} \cup \\ & \{(A, \mathsf{Flow}(R_i)) & \mid A \in N^{\mathrm{out}}(R_i)\} \ . \end{aligned}$$

Finally,

$$\mathcal{F}_{|I|+1} = \{(A, \mathrm{Bal}(A)) \mid A \in J \setminus \mathcal{H}\} \ .$$

Now, we prove that this is really a valid w-Res derivation of $(C, \mathrm{Bal}(C))$. First we have to check that all weights in $\mathcal{F}''_0$ are non-negative. This is because all clauses $A \in \mathcal{H}$ only have outgoing edges, and therefore $\mathrm{Bal}(A) \leq 0$, and the weights in $\mathcal{F}''_0$ are positive. Then multisets $\mathcal{F}''_0$, $\mathcal{F}'_0$ and $\mathcal{F}_0$ are all fold-unfold equivalent and hence valid steps in w-Res.

Now, for each $i \in [I]$ we see the step from $\mathcal{F}_{i-1}$ to $\mathcal{F}_i$ as the application of the rule R_i with weight $\mathsf{Flow}(R_i)$. The premises of R_i are the clauses A such that $A \in N^{\mathrm{in}}(R_i)$ and the conclusions are the clauses A such that $A \in N^{\mathrm{out}}(R_i)$.

The multiset $\mathcal{F}_{|I|}$ then is the following:

$$\begin{aligned}
\mathcal{F}_{|I|} = \ & \mathcal{F}_0 \cup \\
& \{(A, -\mathsf{Flow}(R)) \quad \mid R \in I \wedge A \in N^{\mathrm{in}}(R)\} \cup \\
& \{(A, \mathsf{Flow}(R)) \quad \mid R \in I \wedge A \in N^{\mathrm{out}}(R)\} \\
= \ & \{(A, \mathrm{Bal}(A)) \quad \mid A \in J \setminus \mathcal{H}\} \cup \\
& \{(A, \mathsf{Flow}(R)) \quad \mid A \in J \wedge R \in N^{\mathrm{out}}(A)\} \cup \\
& \{(A, -\mathsf{Flow}(R)) \quad \mid A \in J \wedge R \in N^{\mathrm{in}}(A)\} \cup \\
& \{(A, -\mathsf{Flow}(R)) \quad \mid R \in I \wedge A \in N^{\mathrm{in}}(R)\} \cup \\
& \{(A, \mathsf{Flow}(R)) \quad \mid R \in I \wedge A \in N^{\mathrm{out}}(R)\} \ ,
\end{aligned}$$

Now, trivially, for $R \in I$ and $A \in J$ we have $R \in N^{\mathrm{in}}(A)$ if and only if $A \in N^{\mathrm{out}}(R)$ and, similarly, $R \in N^{\mathrm{out}}(A)$ if and only if $A \in N^{\mathrm{in}}(R)$, therefore

$$\mathcal{F}_{|I|+1} = \{(A, \mathrm{Bal}(A)) \mid A \in J \setminus \mathcal{H}\} \approx \mathcal{F}_{|I|} \ .$$

Moreover, all the weights in $\mathcal{F}_{|I|+1}$ are non-negative and $(C, \mathrm{Bal}(C)) \in \mathcal{F}_{|I|+1}$. □

Notice that the same argument we used in Theorem 4.3 generalizes trivially to a generic set of inference rules, provided that Lemma 3.10 holds and the rules allow having constant size c-Res derivations of clauses from themselves as in Figure 3.

5. Algebraic Proofs as Weighted Proofs

In Section 4 we showed that w-Res and c-Res are equivalent proof systems. In this section we show the connection between weighted proofs and (semi)-algebraic proof systems in particular the Nullstellensatz proof system and the Sherali-Adams proof system.

5.1 Extra Preliminaries

Let X be the set of variables $x_1, \ldots, x_n, \bar{x}_1, \ldots, \bar{x}_n$. The variables in X are intended to be Boolean and the $\bar{x}_i$'s are *new* variables with intended meaning the negation of the x_i's. By $\mathbb{Z}[X]$ we denote the set of polynomials in the variables X and coefficients in $\mathbb{Z}$.

5.1.1 Nullstellensatz

The Nullstellensatz proof system (NS) was introduced by Beame, Impagliazzo, Krajicek, Pitassi, and Pudlak (1994) and it is a way to get a proof system from (the weak form of) Hilbert's Nullstellensatz.

Definition 5.1 (Nullstellensatz, $\mathsf{NS}_\mathbb{Z}$)**.** Given polynomials $p_1, \ldots, p_m, s$ in $\mathbb{Z}[X]$, a *Nullstellensatz* ($\mathsf{NS}_\mathbb{Z}$) *derivation of the equality* $s = 0$ *from the set of equalities* $\{p_1 = 0, \ldots, p_m = 0\}$ is a polynomial identity of the form

$$\sum_{i \in [m]} q_i p_i + \sum_{j=1}^{n} r_j (x_j^2 - x_j) + \sum_{j=1}^{n} r'_j (x_j + \bar{x}_j - 1) = s \ , \tag{4}$$

where $q_i, r_j, r'_j \in \mathbb{Z}[X]$. A *refutation of* $\{p_1 = 0, \ldots, p_m = 0\}$ is a derivation of $c = 0$ where c is a non-zero constant. The *size* of the polynomial identity in (4) is the length of a bit-string representing the polynomials q_i, r_j, r'_j, including the coefficients. If all the coefficients are ± 1 we call the system *unary* NS. The *degree* of the polynomial identity in (4) is

$$\max\{\deg q_i + \deg p_i, \deg r_j + 2, \deg r'_j + 1 \colon i \in [m], j \in [n]\} \ ,$$

where $\deg p$ is the degree of the polynomial p.

Notice that in the left hand side (LHS) of (4) we don't have a sum corresponding to polynomials $\bar{x}_i^2 - \bar{x}_i$. The reason is that they are not needed to enforce the variables $\bar{x}_i$ to be Boolean, indeed:

$$\bar{x}_i^2 - \bar{x}_i = (\bar{x}_i - x_i)(x_i + \bar{x}_i - 1) + (x_i^2 - x_i) \ .$$

NS is sound and complete, i.e. the set of equations $\{p_1 = 0, \ldots, p_m = 0\}$ is unsatisfiable over $\{0, 1\}$-assignments if and only if there is a NS refutation of it.

NS together with an encoding of CNF formulas into sets of polynomials is a propositional proof system. A CNF formula $\{C_1, \ldots, C_m\}$ is encoded as the set of polynomial equations $\{M_{C_i} = 0, \ldots, M_{C_m} = 0\}$, where if $C = \bigvee_{i \in I} x_i \vee \bigvee_{j \in J} \neg x_j$, then M_C is the monomial $\prod_{i \in I} \bar{x}_i \prod_{j \in J} x_j$. This encoding has the property that if a truth-assignment α satisfies the clause C, then α satisfies the equation $M_C = 0$, and if a truth-assignment α falsisfies the clause C, then α satisfies the equation $M_C = 1$. It is not hard to see that unary NS p-simulates tree-like Res. NS is exponentially stronger than tree-like Res since it can prove efficiently the *onto-functional* pigeonhole principle. Interestingly, Bonacina and Bonet (2022) showed that, in some intuitive sense, the principles that NS can prove efficiently, and (tree-like) Res cannot, are just formulas easily reducible to the *onto-functional* pigeonhole principle. Göös et al. (2022) showed that degree-polylog(n) unary NS corresponds to the TFNP class PPAD.

If a CNF formula in n variables has a degree-d NS refutation, then it has a refutation of size $n^{O(d)}$ and coefficients of size $n^{O(1)}$ when expressed in binary. This bound is also asymptotically tight, i.e. there are CNF formulas that can be refuted in degree-d NS but require size $n^{\Omega(d)}$ (Loera et al., 2009).

The main reason we have the formal variables $\bar{x}_i$ is to allow this succinct encoding. Semantically the variable $\bar{x}_i$ is equivalent to the polynomial $1 - x_i$, but encoding the clause $\bigvee_{i \in [n]} x_i$ as the equation $\prod_{i \in [n]} \bar{x}_i = 0$ is very different from encoding it as the polynomial equation $\prod_{i \in [n]} (1 - x_i) = 0$ in terms of number of monomials. In NS we are not allowed to handle arbitrary algebraic expressions like $\prod_{i \in [n]} (1 - x_i)$, but only polynomials, that is sums of monomials. So, once written as a polynomial, $\prod_{i \in [n]} (1 - x_i)$ has an exponential number of monomials.

A variant of NS *without the $\bar{x}_i$ variables* (NS without twin variables) has been considered in the literature. Essentially it amounts to an identity as in (4) where each instance of $\bar{x}_i$ is substituted with $1 - x_i$ and the resulting algebraic expression is expanded as a sum of monomials. This system is not suited to refute arbitrary CNF formulas due to the potential exponential size encoding of CNFs, but it might be used to refute k-CNF formulas for $k = O(\log n)$. de Rezende et al. (2021b) showed that this version of NS without twin variables is exponentially weaker than NS.

Remark 5.2. The Nullstellensatz proof system could be defined for polynomials with coefficients in arbitrary rings, and a special case of interest is polynomials with coefficients in finite fields such as $GF(2)$. The characterization of Nullstellensatz as restricted w-Res (Lemmas 5.13 and 5.11) could be adapted to this more general context. Indeed, while w-Res cannot be generalized to $GF(2)$, *restricted* w-Res could and, it would correspond to Nullstellensatz over $GF(2)$ with minor adaptations of the arguments in Lemmas 5.13 and 5.11. For simplicity we focus on Nullstellensatz and polynomials with integer coefficients.

5.1.2 Sherali-Adams

The Sherali-Adams proof system was introduced by Dantchev (2007) and Dantchev, Martin, and Rhodes (2009a) as a way to get proof systems out of the hierarchy of relaxations introduced by Sherali and Adams (1990) to solve Integer Linear Programs.

Definition 5.3 (Sherali-Adams, SA)**.** Given a set of polynomials $p_1, \dots, p_m$ and a polynomial s in $\mathbb{Z}[X]$, a *Sherali-Adams* ($\mathsf{SA}_{\mathbb{Z}}$) *derivation of $s \geq 0$ from $p_1 \geq 0, \dots, p_m \geq 0$* is a polynomial identity of the form

$$\sum_{i \in [m]} q_i p_i + \sum_{j=1}^{n} r_j (x_j^2 - x_j) + \sum_{j=1}^{n} r'_j (x_j + \bar{x}_j - 1) + q = s \ , \tag{5}$$

where $q, q_i, r_j, r'_j \in \mathbb{Z}[X]$ and additionally q, q_i have only positive coefficients. A *refutation of* $\{p_1 \geq 0, \dots, p_m \geq 0\}$ is a derivation of $-c \geq 0$ for a positive constant c. The *size* of the polynomial identity in (5) is the length of a bit-string representing the polynomials q, q_p, r_j, r'_j, including the coefficients. If all the coefficients are ±1 we call the system *unary* SA. The *degree* of the polynomial identity in (5) is

$$\max\{\deg q_i + \deg p_i, \deg r_j + 2, \deg r'_j + 1, \deg q \colon i \in [m], j \in [n]\} \ ,$$

where $\deg p$ is the degree of the polynomial p.

An example of a SA derivation is in Example 5.8. In this section, we consider only Sherali-Adams over the ring $\mathbb{Z}$, hence we refer to $\mathsf{SA}_{\mathbb{Z}}$ simply as SA, omitting the reference to $\mathbb{Z}$.

Similarly as for NS, in (5) we might want to eliminate the variables $\bar{x}_1, \dots, \bar{x}_n$, and hence consider the algebraic equality in (5) after the substitution $\bar{x}_i$ for $1 - x_i$, for each $i \in [n]$. As for NS, de Rezende et al. (2021b) showed that the resulting system is known to be exponentially weaker than SA.

SA is sound and complete, i.e. the system of inequalities $\{p_1 \geq 0, \dots, p_\ell \geq 0\}$ is unsatisfiable over $\{0, 1\}$-assignments if and only if there is a SA refutation of it. SA together with an encoding of CNF clauses into sets of polynomials is a propositional proof system. We use the same encoding as for NS with the only difference that an equality is encoded as two inequalities, i.e. the clause $C = \bigvee_{i \in I} x_i \vee \bigvee_{j \in J} \neg x_j$ is encoded as the two inequalities

$$\left\{ \prod_{i \in I} \bar{x}_i \prod_{j \in J} x_j \geq 0, \quad -\prod_{i \in I} \bar{x}_i \prod_{j \in J} x_j \geq 0 \right\} .$$

Notice that, the inequality $\prod_{i \in I} \bar{x}_i \prod_{j \in J} x_j \geq 0$ is indeed redundant since the corresponding polynomial could be thought as being part of the polynomial q in (5). Thus we say that the encoding of the clause C is just $-\prod_{i \in I} \bar{x}_i \prod_{j \in J} x_j \geq 0$. We will not use it in this work, but the clause C could also be encoded as the linear inequality $\sum_{i \in I} x_i + \sum_{j \in J} \bar{x}_j \geq 1$.

If a CNF formula in n variables has a degree-d SA refutation, then it has a refutation of size $n^{O(d)}$. See Section 1.2 for more details. This bound is also asymptotically tight, i.e. there are CNF formulas that can be refuted in degree-d SA but require size $n^{\Omega(d)}$ regardless of the degree (Atserias et al., 2016).

Under this encoding of CNF formulas, it is immediate to see that SA p-simulates NS and it is also well-known that SA p-simulates Res, see for instance Lemma 3.5 of (Atserias & Ochremiak, 2018). Moreover, SA is exponentially stronger than Res since it can prove efficiently the pigeonhole principle. Interestingly, Bonacina and Bonet (2022) showed that, in some intuitive sense, what SA can prove efficiently and Res cannot, are just principles reducible to a *weighted* version of the pigeonhole principle. Göös et al. (2022) showed that degree-polylog(n) unary SA corresponds to the TFNP class PPADS.

5.2 Weighted Resolution and NS/SA

We now prove the equivalence between w-Res and SA, and between restricted w-Res and NS. To show that w-Res p-simulates SA we need a simple lemma giving a sort of normal form for SA/NS derivations. This lemma was implicitly used by Bonacina and Bonet (2022) and Fleming et al. (2022). The p-simulation of w-Res by SA generalizes and follows the known simulation of Res by SA.

The next lemmas show that the q_is in the definitions of NS and SA can be taken as positive constants. The first lemma shows that each product $q_i p_i$, from the definitions of NS and SA, can be expressed in a different way.

Lemma 5.4. *Given polynomials $p, q \in \mathbb{Z}[X]$, there are polynomials $r_1, \ldots, r_n$, a polynomial q' in $\mathbb{Z}[X]$ and a non-negative constant a such that*

$$qp = (a - q')p + \sum_{j \in [n]} r_j(x_j + \bar{x}_j - 1) , \tag{6}$$

where q' has all coefficients non-negative. The degrees of q' and $r_j(x_j + \bar{x}_j - 1)$ are at most the degree of q.

Proof. Let bx_jm be a monomial in q, where $b \in \mathbb{Z}$. If $b < 0$ consider the monomial $-bx_jm$ to be part of q'. If $b > 0$ then we can rewrite bmx_jp as

$$bmx_jp = bmp(x_j + \bar{x}_j - 1) - bm\bar{x}_jp + bmp .$$

Now we consider $bmp(x_j + \bar{x}_j - 1)$ as an element of $\sum_j r_j(x_j + \bar{x}_j - 1)$, and the monomial $bm\bar{x}_j$ as part of q'. The procedure is repeated now on the polynomial bmp exhausting all the variables in m one by one.

If the monomial in q is $bm\bar{x}_j$ the identity used is analogous.

At the end we are left with a positive constant a equal to the sum of all positive coefficients b corresponding to monomials in q. □

From this auxiliary lemma we have two normal forms, one for NS and one for SA. Notice that, Atserias and Lauria (2019) also introduce a notion of normal form for SA proofs but this form is not related to the one we introduce here.

Lemma 5.5 (Normal form for NS proofs). *Given a* NS *derivation as in eq.* (4)*, it is possible to transform it, in polynomial time, into a* NS *derivation of the same degree and polynomially larger size, where all the polynomials q_i have the form $a_i - q'_i$, a_i is a positive constant and q'_i has positive coefficients.*

Proof. Immediate from Lemma 5.4. □

Lemma 5.6 (Normal form for SA proofs). *Given a* SA *derivation as in eq.* (5)*, if all the polynomials p_i have negative coefficients then it is possible to transform it, in polynomial time, into a* SA *derivation*

$$\sum_{i\in[m]} a_i p_i + \sum_{j=1}^{n} \Big(r_j(x_j^2 - x_j) + r'_j(x_j + \bar{x}_j - 1) \Big) + q = s \ , \tag{7}$$

where a_i's are positive constants, $q, r_j, r'_j \in \mathbb{Z}[X]$, q has only positive coefficients, the degree is the same of the original derivation and the size is polynomially larger.

Proof. It is immediate from Lemma 5.4:

$$q_i p_i = (a_i - q'_i)p_i + \sum_{j\in[n]} r_{ij}(x_j + \bar{x}_j - 1) \ ,$$

but, by assumption p_i has negative coefficients, hence the polynomial $-q'_i p_i$ has positive coefficients and can be seen as part of the q-part of the SA derivation. □

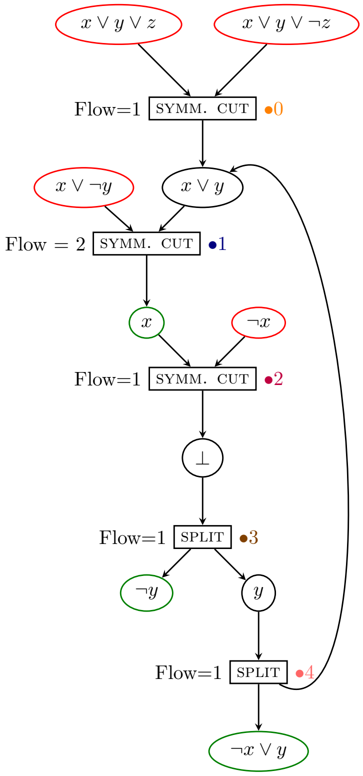
Notice, in particular, this normal form for SA holds when the initial set of polynomials is the encoding of a CNF formula.

Theorem 5.7. SA *is p-equivalent to w*-Res *and* $\mathsf{NS}_{\mathbb{Z}}$ *is p-equivalent to restricted w*-Res*. The same p-equivalences hold between the unary version of the systems. Moreover, degree-d* NS/SA *derivations correspond to width-d w*-Res *derivations.*

By Theorem 4.3, w-Res is p-equivalent to c-Res, and Atserias and Lauria (2019) showed that c-Res is p-equivalent to SA, therefore w-Res and SA are p-equivalent.

We give now a direct and self-contained argument showing the p-equivalence of w-Res and SA that works also in the cases of NS and the systems with unary weights/coefficients. Before proving the equivalence between w-Res and SA, we complete Example 4.4.

Example 5.8 (Example 4.4 cont'd). This is the continuation of Example 4.4 and Figure 2, that is an example of a c-Res/w-Res/SA derivation of $\{x, \neg y, \neg x \vee y\}$ from $\{x \vee y \vee z, x \vee y \vee \neg z, x \vee \neg y, \neg x\}$. In the language of SA, this means the polynomial inequality $-\bar{x} - y - x\bar{y} \geq 0$ is derivable from the set of inequalities $\{-\bar{x}\bar{y}\bar{z} \geq 0, -\bar{x}\bar{y}z \geq 0, -\bar{x}y \geq 0, -x \geq 0\}$. See Figure 4 below.



$$\begin{array}{c l}
(x \vee y \vee z, 1)\ (x \vee y \vee \neg z, 1) & \\
(x \vee \neg y, 2)\ (\neg x, 1) & \\ \hline
 & \text{SYMM.CUT } \& \approx \bullet 0 \\
(x \vee \neg y, 2)\ (\neg x, 1)\ (x \vee y, 1) & \\ \hline
 & \text{SYMM.CUT } \& \approx \bullet 1 \\
(\neg x, 1)\ (x, 2)\ (x \vee y, -1) & \\ \hline
 & \text{SYMM.CUT } \& \approx \bullet 2 \\
(\perp, 1)\ (x, 1)\ (x \vee y, -1) & \\ \hline
 & \text{SPLIT } \& \approx \bullet 3 \\
(y, 1)\ (\neg y, 1)\ (x, 1)\ (x \vee y, -1) & \\ \hline
 & \text{SPLIT } \& \approx \bullet 4 \\
(\neg y, 1)\ (\neg x \vee y, 1)\ (x, 1) &
\end{array}$$

$$\begin{aligned}
-\bar{x}\bar{y}\bar{z} - \bar{x}\bar{y}z - 2\bar{x}y - x \\
&+ \bar{x}\bar{y}(z + \bar{z} - 1) \bullet 0 \\
&+ 2\bar{x}(y + \bar{y} - 1) \bullet 1 \\
&+ (x + \bar{x} - 1) \bullet 2 \\
&- (y + \bar{y} - 1) \bullet 3 \\
&- \bar{y}(x + \bar{x} - 1) \bullet 4 \\
&= -\bar{x} - y - x\bar{y}
\end{aligned}$$

Figure 4: A circular proof, its corresponding weighted proof and the related Sherali-Adams proof.

To prove Theorem 5.7, we adopt a slightly more syntactic view of the clauses in a w-Res derivation: we consider clauses $C = \ell_1 \vee \cdots \vee \ell_r$ where literals might be repeated. In Res, w-Res, and c-Res those clauses are handled implicitly, i.e. identifying $C \vee \ell \vee \ell$ with $C \vee \ell$.

To facilitate the proof of Theorem 5.7, we consider w-Res derivations where this identification is done explicitly via the idempotency rule:

$$\frac{(C \vee \ell \vee \ell, u)}{(C \vee \ell, u)}\textsc{Idempotency} ,$$

where C is a clause and ℓ is a literal and $u \in \mathbb{Z}$. Notice that this rule is redundant since the same conclusion could be obtained using Excluded Middle, Split and Symm. Cut in the following way:

$$
\begin{array}{cl}
(C \lor \ell \lor \ell, u) & \\
\hline
(C \lor \ell \lor \ell, u) \quad (\ell \lor \neg\ell, u) & \textsc{Excluded Middle} \\
\hline
\vdots & \textsc{Split} \\
\hline
(C \lor \ell \lor \ell, u) \quad (C \lor \ell \lor \neg\ell, u) \quad \cdots & \textsc{Split} \\
\hline
(C \lor \ell, u) \quad \cdots & \textsc{Symm. Cut}
\end{array}
$$

For a clause $C = \bigvee_{i \in I} x_i \lor \bigvee_{j \in J} \neg x_j$, let

$$M(C) = \prod_{i \in I} \bar{x}_i \prod_{j \in J} x_j \ ,$$

and vice versa, given a monomial $m = \prod_{i \in I} \bar{x}_i \prod_{j \in J} x_j$, let

$$C(m) = \bigvee_{i \in I} x_i \lor \bigvee_{j \in J} \neg x_j \ .$$

Using this translation, a clause C is encoded as the equality $M(C) = 0$ in NS, and as the inequality $-M(C) \geq 0$ in SA.

Lemma 5.9. *Given multisets of weighted clauses $\mathcal{F}, \mathcal{G}$, we have that $\mathcal{F}$ and $\mathcal{G}$ are fold-unfold equivalent, $\mathcal{F} \approx \mathcal{G}$, if and only if*

$$\sum_{(C,w) \in \mathcal{F}} wM(C) = \sum_{(C,w) \in \mathcal{G}} wM(C) \ .$$

Proof. By definition, $\mathcal{F} \approx \mathcal{G}$ means that for every clause C,

$$\sum_{w:(C,w) \in \mathcal{F}} w = \sum_{w:(C,w) \in \mathcal{G}} w \ ,$$

that is

$$\sum_{C} \left(\sum_{w:(C,w) \in \mathcal{F}} w - \sum_{w:(C,w) \in \mathcal{G}} w \right) M(C) = 0 \ ,$$

and therefore

$$\sum_{(C,w) \in \mathcal{F}} wM(C) = \sum_{(C,w) \in \mathcal{G}} wM(C) \ .$$

Vice versa, if

$$\sum_{(C,w) \in \mathcal{F}} wM(C) = \sum_{(C,w) \in \mathcal{G}} wM(C)$$

it means that

$$\sum_{C}\left(\sum_{w:(C,w)\in\mathcal{F}} w - \sum_{w:(C,w)\in\mathcal{G}} w\right) M(C) = 0 \ ,$$

hence for each $M(C)$ the coefficient in front of it must be 0 and the fact that $\mathcal{F} \approx \mathcal{G}$ follows immediately. □

We split the various p-simulations in Theorem 5.7 into distinct lemmas.

Lemma 5.10. *w-*Res *p-simulates* SA.

Proof. Let $\mathcal{F} = \{C_1, \dots, C_\ell\}$. By Lemma 5.6, without loss of generality, an SA refutation of $\mathcal{F}$ has the form

$$-c = -\sum_{i=1}^{\ell} a_i M(C_i) + \sum_{j=1}^{n} r_j(x_j^2 - x_j) + \sum_{j=1}^{n} r'_j(x_j + \bar{x}_j - 1) + \sum_{i\in J} a'_i m'_i \ , \tag{8}$$

for some positive constants c, a_i, a'_i, polynomials r_j, r'_j, and monomials m'_i. Or, equivalently

$$-c - \sum_{i\in J} a'_i m'_i = -\sum_{i=1}^{\ell} a_i M(C_i) + \sum_{j=1}^{n} r_j(x_j^2 - x_j) + \sum_{j=1}^{n} r'_j(x_j + \bar{x}_j - 1) \ . \tag{9}$$

Notice that the LHS of eq. (9) has only negative coefficients. The constant $-c$ is the weighted clause $(\bot,\ c)$ and $-a_i M(C_i)$ is $(C_i,\ a_i)$.

We construct a w-Res refutation $(\mathcal{L}_1, \dots, \mathcal{L}_s)$ of $\mathcal{F}$. The multiset

$$\mathcal{L}_1 = \{(C_1,\ a_1), \dots, (C_\ell,\ a_\ell)\}$$

corresponds to $-\sum_{i=1}^{\ell} a_i M(C_i)$. Suppose we already constructed $\mathcal{L}_t$, then pick any binomial of the form $am(x_j^2 - x_j)$, not already picked from the sum $\sum_{j=1}^{n} r_j(x_j^2 - x_j)$, and let

$$\mathcal{L}_{t+1} = \mathcal{L}_t \cup \{(C(m) \vee \neg x_j \vee \neg x_j,\ -a),\ (C(m) \vee \neg x_j,\ a)\} \ .$$

This is an application of the Idempotency rule. Continue this way until all the binomials from $\sum_{j=1}^{n} r_j(x_j^2 - x_j)$ are picked. Then continue with the trinomials from $\sum_{j=1}^{n} r'_j(x_j + \bar{x}_j - 1)$. Suppose we constructed $\mathcal{L}_k$, then pick any trinomial of the form $am(x_j + \bar{x}_j - 1)$, not already picked from the sum $\sum_{j=1}^{n} r'_j(x_j + \bar{x}_j - 1)$, and let

$$\mathcal{L}_{k+1} = \mathcal{L}_k \cup \{(C(m) \vee \neg x_j,\ -a),\ (C(m) \vee x_j, -a),\ (C(m),\ a)\} \ .$$

This is an application of a symmetric cut. Continue this process until all the trinomials from $\sum_{j=1}^{n} r'_j(x_j + \bar{x}_j - 1)$ are picked. Then, let $\mathcal{L}_{s'}$ be the multiset we obtained, that is $\mathcal{L}_{s'}$ is constructed by exhausting all the terms from the RHS of eq. (9). By the equality (9) and Lemma 5.9 this gives

$$\mathcal{L}_{s'} \approx \{(\bot,\ c)\} \cup \{(C(m'_i), a'_i) : i \in I\} \ ,$$

where recall that $a_i' \geq 0$. Lemma 5.9 justifies that every clause with negative weight in the constructed proof has to have the same clause with the corresponding positive weight in $\mathcal{L}_{s'}$.

Given the translation from monomials to clauses, we have constructed a w-Res derivation of width equal to the degree of the SA. Moreover, if the coefficients of the SA derivation are ± 1, the weights of the clauses in the w-Res derivation are ± 1 too. □

Lemma 5.11. *Restricted w-Res p-simulates NS.*

Proof. In the case of a NS derivation the argument is analogous to the previous lemma. Let $\mathcal{F} = \{C_1, \dots, C_\ell\}$. A NS refutation of $\{M(C_1) = 0, \dots, M(C_\ell) = 0\}$ has the form

$$c = \sum_{i \in [m]} q_i M(C_i) + \sum_{j=1}^{n} r_j (x_j^2 - x_j) + \sum_{j=1}^{n} r_j' (x_j + \bar{x}_j - 1) \ , \tag{10}$$

where $q_i, r_j, r_j' \in \mathbb{Z}[X]$ and $c \neq 0$. We have two cases $c > 0$ and $c < 0$.

If $c > 0$ we apply the Normal Form of NS (Lemma 5.5) and get

$$c = \sum_{i \in [m]} (a_i - q_i') M(C_i) + \sum_{j=1}^{n} r_j (x_j^2 - x_j) + \sum_{j=1}^{n} r_j' (x_j + \bar{x}_j - 1) \ ,$$

where the a_is are positive and all the q_i's have positive coefficients. We then change sign to both sides of the equality and get

$$-c - \sum_{i \in [m]} q_i' M(C_i) = - \sum_{i \in [m]} a_i M(C_i) + \sum_{j=1}^{n} r_j (x_j^2 - x_j) + \sum_{j=1}^{n} r_j' (x_j + \bar{x}_j - 1) \ .$$

If $c < 0$ we first change sign to both sides of (10) to reduce to the previous case.

Then we argue as in the previous lemma. To conclude, the only new small fact to notice is that each monomial in the sum $\sum_{i \in [m]} q_i' M(C_i)$, is a weakening of an initial axiom, once translated to a weighted clause. □

Lemma 5.12. *SA p-simulates w-Res.*

Proof. We show now how to convert a w-Res refutation into a SA refutation. Let $(\mathcal{L}_1, \dots, \mathcal{L}_t)$ be a w-Res refutation of some multiset of clauses $\mathcal{F}$. We construct algebraic expressions $S_1, \dots, S_t$ with the property that $S_i = -\sum_{(C,\ w) \in \mathcal{L}_i} wM(C)$ and they are valid SA derivations.

Let $S_1 = -\sum_{(C,\ w) \in \mathcal{L}_1} wM(C)$. By assumption, all the clauses C in $\mathcal{L}_1$ are clauses from $\mathcal{F}$. Then suppose we constructed an algebraic expression $S_i = -\sum_{(C,\ w) \in \mathcal{L}_i} wM(C)$ having the form of a SA derivation. We want then to construct S_{i+1}.

- If $\mathcal{L}_i \approx \mathcal{L}_{i+1}$ let $S_{i+1} = S_i$.
- If $\mathcal{L}_{i+1}$ is obtained from $\mathcal{L}_i$ by a Symmetric-Cut rule $\frac{(C \vee x,\ w) \quad (C \vee \neg x,\ w)}{(C,\ w)}$, then add to the algebraic expression S_i the terms

  $$-wM(C) + wM(C \vee x) + wM(C \vee \neg x) = wM(C)(x + \bar{x} - 1) \ .$$

  That is $S_{i+1} = S_i + wM(C)(x + \bar{x} - 1)$.

- If $\mathcal{L}_{i+1}$ is obtained from $\mathcal{L}_i$ by a Split rule $\frac{(C,\ w)}{(C\vee x,\ w) \quad (C\vee\neg x,\ w)}$, then add to S_i the terms
$$wM(C) - wM(C \vee x) - wM(C \vee \neg x) = -wM(C)(x + \bar{x} - 1)\ .$$
That is $S_{i+1} = S_i - wM(C)(x + \bar{x} - 1)$.

- If $\mathcal{L}_{i+1}$ is obtained from $\mathcal{L}_i$ by a Idempotency rule $\frac{(C\vee\neg x\vee\neg x,\ w)}{(C\vee\neg x,\ w)}$ then add to S_i the terms
$$-wM(C \vee \neg x) + wM(C \vee \neg x \vee \neg x) = wM(C)(x^2 - x)\ .$$
That is $S_{i+1} = S_i + wM(C)(x^2 - x)$.

- If $\mathcal{L}_{i+1}$ is obtained from $\mathcal{L}_i$ by an Idempotency rule $\frac{(C\vee x\vee x,\ w)}{(C\vee x,\ w)}$ then add to S_i the terms
$$\begin{aligned} -wM(C \vee x) + wM(C \vee x \vee x) &= wM(C)(\bar{x}^2 - \bar{x}) \\ &= wM(C)(x^2 - x) \\ &\quad + (wM(C)\bar{x} - wM(C)x)(\bar{x} + x - 1)\ . \end{aligned}$$
That is $S_{i+1} = S_i + wM(C)(x^2 - x) + (wM(C)\bar{x} - wM(C)x)(\bar{x} + x - 1)$.

- If $\mathcal{L}_{i+1}$ is obtained from $\mathcal{L}_i$ by an Excluded Middle rule $\frac{}{(x\vee\neg x,\ w)}$ then add to S_i the terms
$$\begin{aligned} -wM(x \vee \neg x) &= -wx\bar{x} \\ &= -wx(x + \bar{x} - 1) + w(x^2 - x)\ . \end{aligned}$$
That is $S_{i+1} = S_i - wx(x + \bar{x} - 1) + w(x^2 - x)$.

It is immediate to see that S_{i+1} constructed above is such that
$$S_{i+1} = -\sum_{(C,\ w)\in\mathcal{L}_{i+1}} wM(C)\ ,$$
and in particular $S_t = -\sum_{(C,\ w)\in\mathcal{L}_t} wM(C)$. Since we started with a w-Res refutation in $\mathcal{L}_t$, we have the weighted clause $(\perp, c)$ for some $c > 0$, and all the other clauses have positive coefficients; therefore, $S_t = -c - q$ where q is a polynomial with positive coefficients. Therefore $S_t + q = -c$ is a SA refutation. □

Lemma 5.13. NS *p-simulates restricted* w-Res*.*

Proof. If instead of a w-Res refutation we had a restricted w-Res refutation, the argument is the same as in the previous lemma, the only difference is that now q is a sum of multiples of initial axioms, therefore $S_s + q = -c$ is a NS refutation. □

6. Examples

In the previous sections we saw some simple examples of SA/w-Res/c-Res derivations in the context of exemplifying the equivalences among the systems. In this section we see two families of contradictions provable in SA that are hard for Res: the graph pigeonhole principle and the Hex principle. Moreover, there are polynomial time algorithms to find their SA proofs.

6.1 The Graph-Pigeonhole Principle

Let p_{ij} with $i \in [m]$ and $j \in [n]$ be the Boolean variables of the pigeonhole principle. With the usual intended meaning that p_{ij} is 1 if the pigeon i flies to the hole j and 0 otherwise. The pigeonhole principle $\mathtt{PHP}^m_n$ is the conjunction of the clauses:

$$\begin{array}{lll} \text{(totality axioms)} & \displaystyle\bigvee_{j \in [n]} p_{ij} & \text{for all } i \in [m] \ , \\ \text{(injectivity axioms)} & \neg p_{ij} \vee \neg p_{i'j} & \text{for all } i \neq i' \in [m] \text{ and for all } j \in [n] \ . \end{array}$$

The encoding of the pigeonhole principle $\mathtt{PHP}^m_n$ as a set of polynomial inequalities is

$$\{-\prod_{j \in [n]} \bar{x}_{ij} \geq 0 \ : \ i \in [m]\} \cup \{-x_{ij}x_{i'j} \geq 0 \ : \ i \neq i' \in [m], \ j \in [n]\} \ .$$

We use Boolean variables x_{ij} for the algebraic encoding and p_{ij} for the propositional encoding just for clarity: $x_{ij} = 1$ if and only if $p_{ij} = 1$. The formula $\mathtt{PHP}^m_n$ is unsatisfiable whenever $m > n$.

Given a bipartite graph G with bipartition (P, H), the *graph* pigeonhole principle $\mathtt{PHP}(G)$ is $\mathtt{PHP}^{|P|}_{|H|}$ once restricted by the assignment mapping $p_{ij} = 0$ for each $(i, j) \notin E(G)$, and for the algebraic encoding $x_{ij} = 0, \bar{x}_{ij} = 1$ for each $(i, j) \notin E(G)$.

It is well known that $\mathtt{PHP}^{n+1}_n$ has polynomial-size SA refutations, but it requires linear degree (see for instance (Rhodes, 2007)) and it is hard for Resolution. The same is true for the bijective pigeonhole principle, which is hard for Resolution but easy for Sherali-Adams (Göös et al., 2022). For completeness, in Appendix A, we give a short proof of $\mathtt{PHP}^m_n$ in the language of SA. Other, seemingly different, proofs of $\mathtt{PHP}^m_n$ were given in the language of c-Res (Atserias & Lauria, 2019, Theorem 4) and in the language of MaxSAT Resolution with Extension by Larrosa and Rollon (2020). It is not hard to see that all those proofs are essentially the same argument seen in different languages.

Proposition 6.1. *Let G be a bipartite graph with bipartition $([m], [n])$ and $m > n$. $\mathtt{PHP}(G)$ has polynomial-size unary SA refutations of degree at most the degree of the graph G.*

A direct consequence of Proposition 6.1 is that such refutations of $\mathtt{PHP}(G)$ can be found in time $n^{O(d)}$, where d is the maximum degree of G.

We know that SA/c-Res/w-Res are able to p-simulate Res and to prove $\mathtt{PHP}(G)$. What other combinatorial principles can be proved efficiently by these systems? Informally, Bonacina and Bonet (2022) show that SA/c-Res/w-Res are able to prove (additionally to what Res is able to prove) only principles reducible to a *weighted* version of $\mathtt{PHP}(G)$.

Notice that not all combinatorial principles reducible to $\mathtt{PHP}^m_n$ are provable efficiently in SA. For instance, the *bit-encoding* of $\mathtt{PHP}^m_n$ requires exponential size refutations in SA/w-Res/c-Res (Dantchev, Ghani, & Martin, 2020).

6.2 Hex

We can have principles that are equivalent to $\mathtt{PHP}(G)$ and efficiently refutable in SA, but such that the equivalence is not at all obvious. One of those principles is *Hex.* Buss (2006) show that a formalization *Hex* and PHP(G) are equivalent modulo bounded-depth Frege reductions but is was not clear that such reduction could be carried out in systems working on clauses such as Resolution or Sherali-Adams (seen as weighted Resolution).

Consider a Hex board of size $n \times n$, i.e. a parallelogram of size $n \times n$ tiled with hexagons and the tiles at the border have colors Blue (B), Cyan (C), Magenta (M), and Red (R). See Fig. 5 for an example of Hex board with $n = 8$.

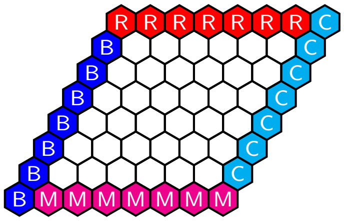


Figure 5: A 8×8 Hex board with borders filled in.

There are two players that take turns coloring the tiles of the board. Player I colors tiles with either Blue or Cyan and Player II colors tiles with either Magenta and Red. The goal of Player I is to create a path consisting only of Blue-Cyan tiles connecting opposite sides of the board. Similarly, the goal of Player II is to create a path consisting only of Magenta-Red tiles connecting opposite sides of the board.

Every completely filled Hex board has a winner, i.e. a path either consisting only of Blue-Cyan tiles or Red-Magenta tiles connecting opposite sides of the board. Intuitively, the $\mathtt{HEX}_n$ formula asserts that there is no winner by forbidding those type of paths, and therefore it is unsatisfiable. We use two pairs of similar colors to express this principle as clauses; the non existence of a path connecting opposite sides of the board is expressed saying that there are no adjacent Red-Magenta or Blue-Cyan tiles.

For the description of the $\mathtt{HEX}_n$ formula we follow (Buss, 2006). Given an hexagon in position (i, j), it is adjacent to the hexagons in positions $(i, j-1)$, $(i, j+1)$, $(i+1, j-1)$, $(i+1, j)$, $(i-1, j-1)$, $(i-1, j+1)$. For each hexagon h we have 4 variables B_h, C_h, M_h, R_h indicating its color (B for Blue, C for Cyan, M for Magenta, R for Red). The formula $\mathtt{HEX}_n$ is then the conjunction of the following clauses:

1. Unit clauses expressing that the borders of the board are pre-colored as in Fig. 5: i.e. the clauses $B_{1,1}, \dots, B_{1,n-1}$, $R_{1,n}, \dots, R_{n-1,n}$, $C_{n,2}, \dots, C_{n,n}$, and $M_{2,1}, \dots, M_{n,1}$.

2. Clauses stating that each hexagon gets exactly one color, i.e. for each hexagon h we have the clauses $B_h \vee C_h \vee M_h \vee R_h$ together with the clauses $\neg B_h \vee \neg C_h$, $\neg B_h \vee \neg M_h$, $\neg B_h \vee \neg R_h$, $\neg C_h \vee \neg M_h$, $\neg C_h \vee \neg R_h$, and $\neg M_h \vee \neg R_h$.

3. Clauses stating that no adjacent hexagons can be colored red-magenta or blue-cyan, i.e. the clauses $\neg R_h \vee \neg M_{h'}$, $\neg B_h \vee \neg C_{h'}$ for each adjacent hexagons h, h'.

The clauses above in the context of SA derivations gets encoded as the following inequalities

1. $-\bar{B}_{1,1} \geq 0, \dots, -\bar{B}_{1,n-1} \geq 0$, $-\bar{R}_{1,n} \geq 0, \dots, -\bar{R}_{n-1,n} \geq 0$, $-\bar{C}_{n,2} \geq 0, \dots, -\bar{C}_{n,n} \geq 0$, and $-\bar{M}_{2,1} \geq 0, \dots, -\bar{M}_{n,1} \geq 0$.

2. For each hexagon h we have the inequalities $-\bar{B}_h\bar{C}_h\bar{M}_h\bar{R}_h \geq 0$ together with the inequalities $-B_hC_h \geq 0$, $-B_hM_h \geq 0$, $-B_hR_h \geq 0$, $-C_hM_h \geq 0$, $-C_hR_h \geq 0$, and $-M_hR_h \geq 0$.

3. The inequalities $-R_hM_{h'} \geq 0$, $-B_hC_{h'} \geq 0$ for each adjacent hexagons h, h'.

Notice that, the inequalities in item 2 are semantically equivalent to

$$B_h + C_h + M_h + R_h = 1 \ ,$$

and this has also short SA derivations from item 2.

Proposition 6.2. $\mathtt{HEX}_n$ *has polynomial-size* $\mathsf{SA}/c\text{-}\mathsf{Res}/w\text{-}\mathsf{Res}$ *refutations.*

The $\mathtt{HEX}_n$ principle looks very different from a pigeonhole principle but indeed it is a pigeonhole principle in disguise, see (Buss, 2006).

Proof. We construct a pigeonhole principle associated to $\mathtt{HEX}_n$. The set of pigeons P are all the vertices in the Hex $n \times n$ board where exactly three tiles meet. The set of holes H are all the vertices in the same Hex board where exactly three tiles meet except the bottom-left vertex (where two Blue tiles and a Magenta tile meet). Each pigeon is only allowed to fly to the hole in the very same place where the pigeon is or to the (at most three) holes adjacent to it. Let G be this bipartite graph of degree at most 4 and let x_{ij} be the variables of $\mathtt{PHP}(G)$ in the polynomial encoding. For $i \in P$, let $N(i)$ be the set of its neighbors in G. Notice that G has a loop in each vertex in $P \setminus H$, in particular, for each $i \in P \setminus H$, $i \in N(i)$.

By Proposition 6.1, we know there is a SA refutation of $\mathtt{PHP}(G)$ of the form

$$\begin{aligned} -1 = -\sum_{i \in P} c_i \prod_{j \in N(i)} \bar{x}_{ij} - & \sum_{\substack{i \neq i' \in P \\ j \in N(i) \cap N(i')}} p_{i,i',j} x_{ij} x_{i'j} \\ & + \sum_{\substack{i \in P \\ j \in N(i)}} (q_{ij}(x_{ij}^2 - x_{ij}) + q'_{ij}(x_{ij} + \bar{x}_{ij} - 1)) \\ & + p \ , \end{aligned} \tag{11}$$

where c_i's are positive constants, q_{ij}, q'_{ij}'s are polynomials, and $p, p_{i,i',j}$'s are polynomials with positive coefficients. Moreover, the size of the refutation is polynomial in $|P| + |Q| = n^{O(1)}$ and p has degree at most 4, the $p_{i,i',j}$'s have degree at most 2, the q_{ij}'s have degree at most 2 and the q'_{ij}'s have degree at most 3.

To get from this a small size refutation of $\mathtt{HEX}_n$ we look at the variables x_{ij} not as indeterminates but as suitable polynomials in the variables of $\mathtt{HEX}_n$. In particular, the polynomials $\prod_{j \in N(i)} \bar{x}_{ij}$, $x_{ij}x_{i'j}$, $x_{ij}^2 - x_{ij}$, and $x_{ij} + \bar{x}_{ij} - 1$ are not axioms anymore but some polynomials we want to derive from the axioms of $\mathtt{HEX}_n$.

For $i \in P$ and $j \in N(i)$, let $l(i,j)$ be the hexagon on the left of the vector $\vec{ij}$ and $r(i,j)$ be the hexagon on the right of the vector $\vec{ij}$.

We want the polynomial $x_{ij} = 1$ if $l(i,j)$ has color Blue and $r(i,j)$ has color Magenta, as in the following figure. In particular, the only vertex in $P \setminus H$ (the one at the bottom

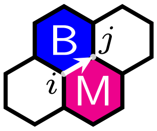


Figure 6: i and j are both pigeons and holes. The arrow shows pigeon i flying to hole j.

left of the Hex board) is mapped to its only neighbour in H. If no pair of tiles adjacent to i is colored Blue/Magenta, we set $x_{ii} = 1$.

In other words, for $i \in P$ and $j \in N(i) \setminus \{i\}$

$$\begin{aligned} x_{ij} &= B_{l(i,j)} M_{r(i,j)} \\ \bar{x}_{ij} &= 1 - x_{ij} \ , \end{aligned}$$

and

$$\begin{aligned} x_{ii} &= \prod_{j \in N(i) \setminus \{i\}} \bar{x}_{ij} \\ &= \prod_{j \in N(i) \setminus \{i\}} (1 - B_{l(i,j)} M_{r(i,j)}) \\ \bar{x}_{ii} &= 1 - x_{ii} \ . \end{aligned}$$

That is, by construction, the polynomials $x_{ij} + \bar{x}_{ij} - 1$ and $x_{ii} + \bar{x}_{ii} - 1$ are identically 0, and therefore derivable from the axioms of $\texttt{HEX}_n$.

First let's see what is the form of the polynomial p in eq. (11) now that it is a polynomial in the variables of $\texttt{HEX}_n$. We use the following identity on the variables $X_1, \dots, X_k$:

$$1 - \prod_{i \in [k]} X_i = \sum_{i \in [k]} \bar{X}_i \prod_{j<i} X_j - \sum_{i \in [k]} (X_i + \bar{X}_i - 1) \prod_{j<i} X_j \ ,$$

to encode the negated variables in low degree.

The polynomial p has degree at most 4 and positive coefficients. Monomials in p in the variables $x_{ij}, \bar{x}_{ij}$, using the identity above, can be transformed into a sum of positive monomials in the variables of $\texttt{HEX}_n$ and multiples of Boolean axioms. Since the degree of p is constant this is just a polynomial increase in the size.

For $j \in N(i)$, the polynomial $x_{ij}^2 - x_{ij}$ has a simple derivation from the Boolean axioms of $\texttt{HEX}_n$:

$$(B_{l(i,j)} M_{r(i,j)})^2 - B_{l(i,j)} M_{r(i,j)} = B_{l(i,j)}^2 (M_{r(i,j)}^2 - M_{r(i,j)}) + M_{r(i,j)} (B_{l(i,j)}^2 - B_{l(i,j)}) \ .$$

Similarly, it is not hard to see that $x_{ii}^2 - x_{ii}$ has a simple derivation from the Boolean axioms of $\texttt{HEX}_n$.

The polynomials $\prod_{j \in N(i)} \bar{x}_{ij}$ are also easy to derive from the Boolean axioms of $\texttt{HEX}_n$ since

$$-\prod_{j \in N(i)} \bar{x}_{ij} = -x_{ii}(1 - x_{ii}) \ .$$

Now consider the polynomials $x_{ij} \cdot x_{i'j}$ of $\texttt{PHP}(G)$. We have two cases.

case 1: i, i', j are distinct vertices such that $j \in N(i) \cap N(i')$, and the corresponding injectivity axiom $x_{ij} \cdot x_{i'j}$ for $\mathtt{PHP}(G)$ is

$$-x_{ij} \cdot x_{i'j} = -B_{l(i,j)} \cdot M_{r(i,j)} \cdot B_{l(i',j)} \cdot M_{r(i',j)} \ .$$

Intuitively, this follows from $\mathtt{HEX}_n$ since if the injectivity was violated then there would be a tile colored both Blue and Magenta, which is not allowed. More formally, either $l(i,j) = r(i',j)$ or $r(i,j) = l(i',j)$ and $x_{ij} \cdot x_{i'j}$ is a multiple of an axiom of $\mathtt{HEX}_n$, an axiom stating that hexagons cannot be colored both Blue and Magenta.

case 2: $i' \in N(i) \setminus \{i\}$, and the corresponding injectivity axiom $x_{i'i} \cdot x_{ii}$ for $\mathtt{PHP}(G)$ is

$$-x_{i'i} \cdot x_{ii} = -B_{l(i',i)} \cdot M_{r(i',i)} \prod_{j \in N(i) \setminus \{i\}} (1 - B_{l(i,j)} M_{r(i,j)})) \ .$$

Intuitively if the pigeon i' flies to i, this means that the tile $l(i',i)$ has color Blue and the tile $r(i',i)$ has color Magenta. Then, the third tile incident to i, (1) must have some color, and (2) that color cannot be Cyan or Red. Therefore, its color must be Blue or Magenta and therefore i does not fly to i. Let $l = l(i'i)$, $r = r(i',i)$ and h be the third hexagon adjacent to i. Then

$$-x_{i'i} \cdot x_{ii} = -B_l \cdot M_r (1 - B_l M_h)(1 - B_h M_r)(1 - B_r M_l) \ .$$

We show this is derivable by a case analysis based on the color of h. The inequalities

$$\begin{aligned}
C_h(-B_l \cdot M_r (1 - B_l M_h)(1 - B_h M_r)(1 - B_r M_l)) &\geq 0 \\
B_h(-B_l \cdot M_r (1 - B_l M_h)(1 - B_h M_r)(1 - B_r M_l)) &\geq 0 \\
R_h(-B_l \cdot M_r (1 - B_l M_h)(1 - B_h M_r)(1 - B_r M_l)) &\geq 0 \\
M_h(-B_l \cdot M_r (1 - B_l M_h)(1 - B_h M_r)(1 - B_r M_l)) &\geq 0 \ ,
\end{aligned}$$

once expanded as sum of monomials, are all derivable from $\mathtt{HEX}_n$. Therefore, summing the previous derivations,

$$(C_h + B_h + R_h + M_h)(-B_l \cdot M_r (1 - B_l M_h)(1 - B_h M_r)(1 - B_r M_l)) \geq 0$$

is also derivable from $\mathtt{HEX}_n$, and since $C_h + B_h + R_h + M_h = 1$, then

$$-B_l \cdot M_r (1 - B_l M_h)(1 - B_h M_r)(1 - B_r M_l) \geq 0$$

is also derivable. □

A direct consequence of the proof of Proposition 6.2 and the degree-automatability of SA is that we can find SA refutations of $\mathtt{HEX}_n$ in polynomial time in n using linear programming.

Acknowledgments

This work was supported by the Ministerio de Ciencia e Innovación/Agencia Estatal de Investigación MCIN/AEI/10.13039/501100011033, Spain [grant numbers PID2019-109137GB-C21, PID2019-109137GB-C22, IJC2018-035334-I, PID2022-138506NB-C21].

Appendix A. Proof of the Pigeonhole Principle in SA

We prove Proposition 6.1. That is, given G be a bipartite graph with bipartition $([m],[n])$ and $m>n$, $\mathtt{PHP}(G)$ has polynomial-size unary SA refutations of degree at most the degree of the graph G.

First we consider $\mathtt{PHP}^m_n$.

Given $j\in[n]$ and $k\in[m]$ let $H_{kj}=\prod_{\ell\in[k]}\bar{x}_{\ell j}$ and $H_{0j}:=1$. Notice that $H_{k+1,j}=\bar{x}_{k+1,j}H_{kj}$.

To improve readability, we highlight in yellow monomials with positive coefficients, in blue Boolean axioms, in red injectivity axioms, and in green totality axioms.

For $j\in[n]$ we have the following equality by a telescopic sum

$$1-\sum_{i\in[m]}x_{ij}=H_{m,j}-\sum_{i\in[m]}\Big(H_{i-1,j}(x_{ij}+\bar{x}_{ij}-1)+\underline{x_{ij}-x_{ij}H_{i-1,j}}\Big)\ ,$$

where the underlined a polynomial has a simple SA derivation from $\mathtt{PHP}^m_n$

$$\underline{x_{ij}-x_{ij}H_{i-1,j}}=\sum_{\ell\in[i-1]}\Big(x_{ij}H_{\ell-1,j}(x_{\ell j}+\bar{x}_{\ell j}-1)-H_{\ell-1,j}\,x_{ij}x_{\ell j}\Big)\ .$$

That is, for each $j\in[n]$ we have the following SA derivation of $1-\sum_{i\in[m]}x_{ij}$ from $\mathtt{PHP}^m_n$:

$$\begin{aligned}1-\sum_{i\in[m]}x_{ij}=H_{m,j}&-\sum_{i\in[m]}H_{i-1,j}(x_{ij}+\bar{x}_{ij}-1)\\&+\sum_{\substack{i\in[m]\\ \ell\in[i-1]}}\Big(x_{ij}H_{\ell-1,j}(x_{\ell j}+\bar{x}_{\ell j}-1)-H_{\ell-1,j}\,x_{ij}x_{\ell j}\Big)\ .\end{aligned}\tag{12}$$

Let π_j be the RHS of eq. (12).

Similarly, for $i\in[m]$ and $k\in[n]$ let $P_{ik}=\prod_{\ell\in[k]}\bar{x}_{i\ell}$. Let $P_{i0}:=1$. As before, $P_{i,k+1}=\bar{x}_{i,k+1}P_{ik}$ and we have

$$\sum_{j\in[n]}x_{ij}-1=-P_{i,n}+\sum_{j\in[n]}\Big(P_{i,j-1}(x_{ij}+\bar{x}_{ij}-1)+\underline{x_{ij}-x_{ij}P_{i,j-1}}\Big)\ ,$$

where the underlined polynomial has a simple SA derivation from $\mathtt{PHP}^m_n$

$$\underline{x_{ij}-x_{ij}P_{i,j-1}}=\sum_{\ell\in[j-1]}\Big(-x_{ij}P_{i,\ell-1}(x_{i\ell}+\bar{x}_{i\ell}-1)+P_{i,\ell-1}x_{ij}x_{i\ell}\Big)$$

That is, for each $i\in[m]$ we have the following SA derivation of $\sum_{j\in[n]}x_{ij}-1$ from $\mathtt{PHP}^m_n$

$$\begin{aligned}\sum_{j\in[n]}x_{ij}-1=-P_{i,n}&+\sum_{j\in[n]}P_{i,j-1}(x_{ij}+\bar{x}_{ij}-1)\\&+\sum_{\substack{j\in[n]\\ \ell\in[j-1]}}\Big(-x_{ij}P_{i,\ell-1}(x_{i\ell}+\bar{x}_{i\ell}-1)+P_{i,\ell-1}x_{ij}x_{i\ell}\Big)\ .\end{aligned}\tag{13}$$

Let ρ_i be the RHS of eq. (13). We have that

$$\sum_{j\in[n]} \pi_j + \sum_{i\in[m]} \rho_i = \sum_{j\in[n]} \left(1 - \sum_{i\in[m]} x_{ij}\right) + \sum_{i\in[m]} \left(\sum_{j\in[n]} x_{ij} - 1\right) = n - m < 0 \ ,$$

and therefore $\sum_{j\in[n]} \pi_j + \sum_{i\in[m]} \rho_i = n - m$ is a SA refutation of $\mathtt{PHP}^m_n$.

By inspecting the previous proof it is immediate to see that for $\mathtt{PHP}(G)$ the degree of the refutation is the maximum degree of the graph G, indeed $\mathtt{PHP}(G)$ is the result of restricting $\mathtt{PHP}^m_n$ mapping some variables x_{ij} to 0. Doing this same restriction to the proof above has the effect of shrinking its degree to the maximum degree of a vertex in G.

References

Ansótegui, C., & Levy, J. (2021). Reducing SAT to Max2SAT. In *Proc. of the 30th International Joint Conference on Artificial Intelligence (IJCAI'21)*, pp. 1367–1373.

Atserias, A., & Hakoniemi, T. (2019). Size-degree trade-offs for sums-of-squares and positivstellensatz proofs. In *Proc. of the 34th Computational Complexity Conference (CCC'19)*, Vol. 137, pp. 24:1–24:20.

Atserias, A., & Lauria, M. (2019). Circular (yet sound) proofs. In *Proc. of the 22nd Int. Conf. on Theory and Applications of Satisfiability Testing (SAT'19)*, Vol. 11628, pp. 1–18.

Atserias, A., Lauria, M., & Nordström, J. (2016). Narrow proofs may be maximally long. *ACM Trans. Comput. Logic*, *17*(3), 19:1–19:30.

Atserias, A., & Müller, M. (2019). Automating resolution is NP-hard. In *Proc. of the 60th IEEE Annual Symp. on Foundations of Computer Science (FOCS'19)*, pp. 498–509.

Atserias, A., & Müller, M. (2020). Automating resolution is NP-hard. *J. ACM*, *67*(5).

Atserias, A., & Ochremiak, J. (2018). Proof complexity meets algebra. *ACM Trans. Comput. Logic*, *20*(1), 1–46.

Beame, P., Impagliazzo, R., Krajicek, J., Pitassi, T., & Pudlak, P. (1994). Lower bounds on Hilbert's Nullstellensatz and propositional proofs. In *Proc. of the 35th Annual Symposium on Foundations of Computer Science (FOCS'94)*, pp. 794–806.

Beame, P., & Pitassi, T. (1996). Simplified and improved resolution lower bounds. In *Proc. of the 37th Annual Symposium on Foundations of Computer Science (FOCS'96)*, pp. 274–282.

Bonacina, I., & Bonet, M. L. (2022). On the strength of Sherali-Adams and Nullstellensatz as propositional proof systems. In *Proc. of the 37th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'22)*, pp. 25:1–25:12.

Bonet, M. L., Buss, S., Ignatiev, A., Marques-Silva, J., & Morgado, A. (2018). Maxsat resolution with the dual rail encoding. In *Proc. of the 32nd AAAI Conference on Artificial Intelligence (AAAI'18)*, pp. 6565–6572.

Bonet, M. L., Buss, S., Ignatiev, A., Morgado, A., & Marques-Silva, J. (2021). Propositional proof systems based on maximum satisfiability. *Artificial Intelligence*, *300*, 103552.

Bonet, M. L., & Levy, J. (2020). Equivalence between systems stronger than resolution. In *Proc. of the 23nd Int. Conf. on Theory and Applications of Satisfiability Testing (SAT'20)*, pp. 166–181.

Bonet, M. L., Levy, J., & Manyà, F. (2006). A complete calculus for Max-SAT. In *Proc. of the 9th Int. Conf. on Theory and Applications of Satisfiability Testing (SAT'06)*, pp. 240–251.

Bonet, M. L., Levy, J., & Manyà, F. (2007). Resolution for Max-SAT. *Artificial Intelligence*, *171*(8-9), 606–618.

Bonet, M. L., Pitassi, T., & Raz, R. (2000). On interpolation and automatization for Frege systems. *SIAM J. Comput.*, *29*(6), 1939–1967.

Bonet, M., Pitassi, T., & Raz, R. (1997). No feasible interpolation for TC^0-Frege proofs. In *Proc. of the 38th Annual Symposium on Foundations of Computer Science (FOCS'97)*, pp. 254–263.

Buss, S. R. (2006). Polynomial-size Frege and resolution proofs of st-connectivity and Hex tautologies. *Theoretical Computer Science*, *357*(1), 35–52.

Cherif, M. S., Habet, D., & Py, M. (2022). From crossing-free resolution to max-sat resolution. In Solnon, C. (Ed.), *28th International Conference on Principles and Practice of Constraint Programming, CP 2022, July 31 to August 8, 2022, Haifa, Israel*, Vol. 235 of *LIPIcs*, pp. 12:1–12:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Cooper, M. C., de Givry, S., & Schiex, T. (2007). Optimal soft arc consistency. In Veloso, M. M. (Ed.), *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pp. 68–73.

Dantchev, S., Martin, B., & Rhodes, M. (2009a). Tight rank lower bounds for the Sherali-Adams proof system. *Theoretical Computer Science*, *410*(21-23), 2054–2063.

Dantchev, S., Martin, B., & Rhodes, M. (2009b). Tight rank lower bounds for the Sherali–Adams proof system. *Theoretical Computer Science*, *410*(21), 2054–2063.

Dantchev, S. S. (2007). Rank complexity gap for lovász-schrijver and sherali-adams proof systems. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '07, p. 311–317, New York, NY, USA. Association for Computing Machinery.

Dantchev, S. S., Ghani, A., & Martin, B. (2020). Sherali-Adams and the binary encoding of combinatorial principles. In *Proc. of the 14th Latin American Symposium (LATIN'20)*, Vol. 12118, pp. 336–347.

de Rezende, S. F., Göös, M., Nordström, J., Pitassi, T., Robere, R., & Sokolov, D. (2021a). Automating algebraic proof systems is NP-hard. In *Proc. of the 53rd Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pp. 209–222. ACM.

de Rezende, S. F., Lauria, M., Nordström, J., & Sokolov, D. (2021b). The power of negative reasoning. In *Proc. of the 36th Computational Complexity Conference (CCC'21)*, Vol. 200, pp. 40:1–40:24.

Filmus, Y., Mahajan, M., Sood, G., & Vinyals, M. (2020). MaxSAT resolution and subcube sums. In *Proc. of the 23nd Int. Conf. on Theory and Applications of Satisfiability Testing (SAT'20)*, pp. 295–311.

Fleming, N., Göös, M., Grosser, S., & Robere, R. (2022). On semi-algebraic proofs and algorithms. In *Proc. of the 13th Innovations in Theoretical Computer Science Conference (ITCS'22)*, Vol. 215, pp. 69:1–69:25.

Fleming, N., Kothari, P., & Pitassi, T. (2019). Semialgebraic proofs and efficient algorithm design. *Found. Trends Theor. Comput. Sci.*, *14*(1-2), 1–221.

Göös, M., Hollender, A., Jain, S., Maystre, G., Pires, W., Robere, R., & Tao, R. (2022). Separations in proof complexity and TFNP. In *Proc. of the 63rd IEEE Annual Symposium on Foundations of Computer Science (FOCS'22)*.

Haken, A. (1985). The intractability of resolution. *Theoretical Computer Science*, *39*, 297–308.

Ignatiev, A., Morgado, A., & Marques-Silva, J. (2017). 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)*, pp. 164–183.

Larrosa, J., & Heras, F. (2005). Resolution in Max-SAT and its relation to local consistency in weighted CSPs. In *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI'05)*, pp. 193–198.

Larrosa, J., & Rollon, E. (2020). 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)*, pp. 218–232.

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

Loera, J. A., Lee, J., Margulies, S., & Onn, S. (2009). Expressing combinatorial problems by systems of polynomial equations and Hilbert's Nullstellensatz. *Combinatorics, Probability and Computing*, *18*(4), 551–582.

Morgado, A., Ignatiev, A., Bonet, M. L., Marques-Silva, J., & Buss, S. (2019). DRMaxSAT with MaxHS: First contact. In *Proc. of the 22nd Int. Conf. on Theory and Applications of Satisfiability Testing (SAT'19)*, Vol. 11628, pp. 239–249.

Py, M., Cherif, M. S., & Habet, D. (2022). Proofs and certificates for max-sat. *J. Artif. Intell. Res.*, *75*, 1373–1400.

Rhodes, M. (2007). Rank lower bounds for the Sherali-Adams operator. In Cooper, S. B., Löwe, B., & Sorbi, A. (Eds.), *Computation and Logic in the Real World*, pp. 648–659, Berlin, Heidelberg. Springer Berlin Heidelberg.

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

Sherali, H. D., & Adams, W. P. (1990). A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM Journal on Discrete Mathematics*, *3*(3), 411–430.

Sherali, H. D., & Adams, W. P. (1994). A hierarchy of relaxations and convex hull characterizations for mixed-integer zero—one programming problems. *Discrete Applied Mathematics*, *52*(1), 83–106.