

Circular (Yet Sound) Proofs

Albert Atserias

Universitat Politècnica de Catalunya

Massimo Lauria

Sapienza - Università di Roma

February 15, 2018

Abstract

We introduce a new way of composing proofs in rule-based proof systems that generalizes tree-like and dag-like proofs. In the new definition, proofs are directed graphs of derived formulas, in which cycles are allowed as long as every formula is derived at least as many times as it is required as a premise. We call such proofs circular. We show that, for all sets of standard inference rules, circular proofs are sound. For Frege we show that circular proofs can be converted into tree-like ones with at most polynomial overhead. For Resolution the translation can no longer be a Resolution proof because, as we show, the pigeonhole principle has circular Resolution proofs of polynomial size. Surprisingly, as proof systems for deriving clauses from clauses, Circular Resolution turns out to be equivalent to Sherali-Adams, a proof system for reasoning through polynomial inequalities that has linear programming at its base. As corollaries we get: 1) polynomial-time (LP-based) algorithms that find circular Resolution proofs of constant width, 2) examples that separate circular from dag-like Resolution, such as the pigeonhole principle and its variants, and 3) exponentially hard cases for circular Resolution.

1 Introduction

In rule-based proof systems, proofs are traditionally presented as sequences of formulas, where each formula is either a hypothesis, or follows from some previous formulas in the sequence by one of the inference rules. Equivalently, such a proof can be represented by a directed acyclic graph, or *dag*, with one vertex for each formula in the sequence, and edges pointing forward from the premises to the conclusions. In this paper we introduce a new way of composing proofs: we allow cycles in this graph as long as every formula is derived at least as many times as it is required as a premise, and show that this structural condition is enough to guarantee soundness. Such proofs we call *circular*.

More formally, our definition is phrased in terms of *flow assignments*: each rule application must carry a positive integer, its *flow* or *multiplicity*, which intuitively means that in order to produce that many copies of the conclusion of the rule we must have produced at least that many copies of each of the premises first. Flow assignments induce a notion of *balance* of a formula in the proof, which is the difference between the number of times that the formula is produced as a conclusion and the number of times that it is required as a premise. Given these definitions, a proof-graph will be a valid circular proof if it admits a flow assignment that satisfies the following *flow-balance* condition: the only formulas of strictly negative balance are

the hypotheses, and the formula that needs to be proved displays strictly positive balance. With this interpretation of flows, circular proofs have the appealing flavour of a network in which demands are fulfilled by the hypotheses, and flow towards the conclusions, which produce surplus. Accordingly, and in analogy with the theory of classical network flows, it makes no difference whether the flows are required to be integers or real numbers, and valid flow assignments can be found efficiently, when they exist, by linear programming techniques.

While proof-graphs with unrestricted cycles are, in general, unsound, we show that circular proofs *are* sound. We offer two very different proofs of this fact. The first one is combinatorial in nature and is phrased in the style of traditional soundness proofs in standard proof systems. Concretely, given a truth assignment that falsifies the conclusion, the soundness proof constructs a path of falsified formulas until it reaches a hypothesis, and does so by induction on the total flow-sum of the flow assignment that satisfies the flow-balance condition. The second proof is (semi-)algebraic and is phrased in the style of the duality theorem for linear programming. Concretely, we phrase the existence of a flow assignment that satisfies the flow-balance condition as the feasibility of a linear program, and observe that the infeasibility of its dual witnesses the soundness of the proof.

Proof complexity of circular proofs With all the definitions in place, we proceed to studying the power of circular proofs from the perspective of propositional proof complexity. For Frege systems, which operate with arbitrary propositional formulas through the standard textbook inference rules, we show that circularity adds no power: the circular, dag-like and tree-like variants of Frege polynomially simulate one another. The equivalence between the dag-like and tree-like variants of Frege is well-known [16]; here we add the circular variant to the list. We prove this by formalizing the LP-based proof of soundness for circular Frege within tree-like Frege itself. To achieve this we make strong use of the formalization of linear arithmetic in Frege that was developed by Buss in order to get efficient Frege proofs of the pigeonhole principle [9], and that was developed further by Goerdts for showing that tree-like Frege simulates the Cutting Planes proof system [13].

For Resolution, we show that circularity *does* make a real difference. First we show that the standard propositional formulation of the pigeonhole principle has circular Resolution proofs of polynomial size. This is in sharp contrast with the well-known fact that Resolution *cannot count*, and that the pigeonhole principle is exponentially hard for (tree-like and dag-like) Resolution [15]. Second we observe that the LP-based proof of soundness of circular Resolution can be formalized in the Sherali-Adams proof system (with twin variables), which is a proof system for reasoning with polynomial inequalities that has linear programming at its base. Sherali-Adams was originally conceived as a hierarchy of linear programming relaxations for integer programs, but it has also been studied from the perspective of proof complexity in recent years.

Surprisingly, it turns out that the converse simulation is also true! For deriving clauses from clauses, Sherali-Adams proofs translate efficiently into circular Resolution proofs. Moreover, both translations, the one from circular Resolution into Sherali-Adams and its converse, are efficient in terms of their natural parameters: length/size and width/degree. As corollaries we obtain for Circular Resolution all the proof complexity-theoretic properties that are known to hold for Sherali-Adams: 1) a polynomial-time (LP-based) proof search algorithm for proofs of bounded width, 2) length-width relationships, 3) separations from dag-like length and width, and 4) explicit exponentially hard examples.

Earlier work While the idea of allowing cycles in proofs is not new, all the instances from the literature that we are aware of are designed for reasoning about inductive definitions, and not for propositional logic, nor for arbitrary inference-based proofs.

Niwiński and Walukiewicz [18] introduced an infinitary tableau method for the modal μ -calculus. The proofs are regular infinite trees that are represented by finite graphs with cycles, along with a decidable *progress condition* on the cycles to guarantee their soundness. A sequent calculus version of this tableau method was proposed in [12], and explored further in [22]. In his PhD thesis, Brotherston [7] introduced a *cyclic* proof system for the extension of first-order logic with inductive definitions; see also [8] for a journal article presentation of the results. The proofs in [8] are ordinary proofs of the first-order sequent calculus extended with the rules that define the inductive predicates, along with a set of *backedges* that link equal formulas in the proof. The soundness is guaranteed by an additional *infinite descent condition* along the cycles that is very much inspired by the progress condition in Niwiński-Walukiewicz’ tableau method. We refer the reader to Section 8 from [8] for a careful overview of the various flavours of proofs with cycles for logics with inductive definitions. From tracing the references in this body of the literature, and as far as we know, it seems that our flow-based definition of circular proofs had not been considered before.

The Sherali-Adams hierarchy of linear programming relaxations has received considerable attention in recent years for its relevance to combinatorial optimization and approximation algorithms; see the original [21], and [4] for a recent survey. In its original presentation, the Sherali-Adams hierarchy can already be thought of as a proof system for reasoning with polynomial inequalities, with the levels of the hierarchy corresponding to the degrees of the polynomials. For propositional logic, the system was studied in [10], and developed further in [20, 3]. Those works consider the version of the proof system in which each propositional variable X comes with a formal *twin variable* \bar{X} , that is to be interpreted by the negation of X . This is the version of Sherali-Adams that we use. It was already known from [11] that this version of the Sherali-Adams proof system polynomially simulates standard Resolution, and has polynomial-size proofs of the pigeonhole principle.

2 Preliminaries

2.1 Formulas

A *literal* is a variable X or the negation of a variable \bar{X} ; we say that \bar{X} is the *complementary literal* of X , and vice-versa. The class of formulas in *negation normal form* is the smallest class of formulas that contains the literals and is closed under binary conjunction \wedge and binary disjunction \vee . A *truth-assignment* is a mapping that assigns a truth-value *true* (1) or *false* (0) to each variable. Truth-assignments evaluate formulas in the natural way through the standard interpretations of negation, conjunction, and disjunction. If a truth-assignment evaluates a formula to true we say that it satisfies it. A *substitution* is a mapping that assigns a formula to each variable. Applying a substitution to a formula means replacing all variables by the formulas to which they are mapped to by the substitution, simultaneously all at once.

We think of disjunction as associative, commutative and idempotent by default, so the formula $(A \vee B) \vee C$ is considered the same as $A \vee (B \vee C)$, which we just write as $A \vee B \vee C$. Also the formula $A \vee B$ is considered the same as $B \vee A$, and the formula $A \vee A$ is considered the same as A . Similarly, we think of conjunction as associative, commutative and idempotent by

default. If A is a formula in negation normal form, we write \overline{A} for its dual formula, which is defined recursively as follows: If A is a literal, then \overline{A} is its complementary literal. If $A = B \vee C$, then $\overline{A} = \overline{B} \wedge \overline{C}$. If $A = B \wedge C$, then $\overline{A} = \overline{B} \vee \overline{C}$. Note that the dual of the dual of A is A itself. The *empty formula* is denoted 0 and is always false by convention. Its complement $\overline{0}$ is denoted 1 , and is always true by convention. We think of 0 and 1 as the neutral elements of \vee and \wedge , respectively, and the absorbing elements of \wedge and \vee , respectively. Thus we view the formulas $0 \vee A$ and $1 \wedge A$ as literally the same as A , and $0 \wedge A$ and $1 \vee A$ as literally the same as 0 and 1 , respectively. The *size* $s(A)$ of a formula A is defined inductively: if A is 0 or 1 , then $s(A) = 0$; if A is a literal, then $s(A) = 1$; if A is a conjunction $C \wedge D$ or a disjunction $C \vee D$ with non-absorbing and non-neutral C and D , then $s(A) = s(C) + s(D) + 1$.

An *elementary tautology* is a formula of the form $\overline{A} \vee A$, where A is a formula. Note that by the convention to view disjunctions and conjunctions as associative, commutative and idempotent, the formula $\overline{A} \vee \overline{B} \vee (A \wedge B)$ is an elementary tautology. If Γ is a set of formulas, a disjunction of formulas in Γ is a formula of the form $A_1 \vee \cdots \vee A_m$, where m is a non-negative integer and each A_i is a formula in Γ . Disjunctions of formulas in Γ are also called Γ -clauses or Γ -cedents. A clause is a disjunction of literals.

2.2 Inference-Based Proofs

An *inference rule* is given by a sequence of *antecedent formulas* A_1, \dots, A_r and a sequence of *consequent formulas* B_1, \dots, B_s with the property that every truth assignment that satisfies all the antecedent formulas also satisfies all the consequent formulas. Here are four important examples:

$$\frac{}{A \vee \overline{A}} \quad \frac{C \vee A \quad D \vee \overline{A}}{C \vee D} \quad \frac{C \vee A \quad D \vee B}{C \vee D \vee (A \wedge B)} \quad \frac{C}{C \vee D}. \quad (1)$$

These inference rules are called *axiom*, *cut*, *introduction of conjunction*, and *weakening*, respectively.

In almost all classical examples in the literature, inference rules have a single consequent formula. The reason for this is that for classical (i.e., non-circular) proofs one may simply split a rule with s consequent formulas into s different single-consequent rules, with little conceptual change. However, for circular proofs a specific rule with two consequent formulas will play an important role; this is the *symmetric weakening*, or *split*, rule:

$$\frac{C}{C \vee A \quad C \vee \overline{A}}. \quad (2)$$

In all these examples the formulas C , D , and A could be the empty formula 0 or its complement 1 . An *instance* of an inference rule is obtained from applying a substitution to its variables. Note that every instance of a rule is a rule itself, which has its own antecedent and consequent formulas.

Fix a set \mathcal{R} of inference rules, a set A_1, \dots, A_m of *hypothesis formulas*, and a *goal* or *apodosis formula* A . A *proof of A from A_1, \dots, A_m* is a finite sequence of formulas that ends in A and such that each formula in the sequence is either contained in A_1, \dots, A_m , or is one of the consequent formulas of an instance of an inference rule in \mathcal{R} that has all its antecedent formulas appearing earlier in the sequence. A *refutation of A_1, \dots, A_m* is a proof of the empty formula 0

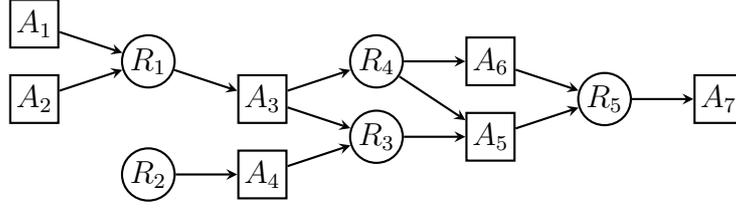


Figure 1: The directed acyclic graph representation of a proof of A_7 from the set of hypothesis formulas A_1 and A_2 through the inference rules R_1, \dots, R_5 . Formula-vertices are represented by boxes and inference-vertices are represented by circles. Formula A_3 is used twice as the antecedent of an inference, and A_5 is produced twice as the consequent of an inference. All rules except R_4 have exactly one consequent formula; R_4 has two. All rules except R_2 have at least one antecedent formula; R_2 has none.

from A_1, \dots, A_m . The *length* of the proof is the length of the sequence, and its *size* is the sum of the sizes of the formulas in the sequence.

Proofs are naturally represented through directed acyclic graphs, a.k.a. *dags*; see Figure 1. The graph has one *formula-vertex* for each formula in the sequence, and one *inference-vertex* for each inference step that produces a formula in the sequence. Each formula-vertex is labelled by the corresponding formula, and each inference-vertex is labelled by the corresponding instance of the corresponding inference rule. Each inference-vertex that is labelled by an inference rule that has r antecedent formulas and s consequent formulas has, accordingly, r incoming edges from the corresponding antecedent formula-vertices, and at least one and at most s outgoing edges towards the corresponding consequent formula-vertices. The directed acyclic graph of a proof Π is its *proof-graph*, and is denoted $G(\Pi)$. A proof Π is called *tree-like* if $G(\Pi)$ is a tree.

2.3 Frege and Resolution Proof Systems

An inference-based proof system is given by a set of allowed inference rules, a set of allowed formulas, and a set of allowed proof-graphs. Two typically sets of allowed proof-graphs are the set of dags, for *dag-like* proofs, and the set of trees, for *tree-like* proofs. If the set of allowed proof-graphs is omitted, dag-like is assumed by default. A proof system P is said to *polynomially simulate* another proof system P' if there is a polynomial-time algorithm that, given a proof Π' in P' as input, computes a proof Π in P , such that Π has the same goal formula and the same hypothesis formulas as Π' . *Frege* and *Resolution* are both inference-based proof systems, as defined next.

In our definition of Frege the set of allowed inference rules are axiom, cut, introduction of conjunction, and weakening as defined in (1), and the set of allowed formulas is the set of all formulas in negation normal form. Frege is sound and (implicational) complete for formulas in negation normal form. This means that if A has a Frege proof from the set of hypothesis formulas A_1, \dots, A_m , then every truth assignment that satisfies all the formulas in A_1, \dots, A_m also satisfies A , and vice-versa.

In our definition of Resolution the only allowed inference rule is cut and the allowed formulas are the clauses. This proof system is sound and complete as a refutation system. This means that

if the set of clauses A_1, \dots, A_m has a Resolution refutation, then there is no truth-assignment that satisfies all clauses A_1, \dots, A_m simultaneously, and vice-versa. In order to turn Resolution into a proof system that is sound and complete for deriving clauses from clauses, one needs to add the axiom and weakening rules to the set of allowed rules. The *width* of a Resolution proof is the number of literals of its largest clause.

2.4 Frege and Resolution with Symmetric Rules

Consider an inference-based proof system in which elementary tautologies of the form $A \vee \overline{A}$ may be introduced at any point in the proof through the axiom rule, and that in addition has the following two nicely symmetric-looking inference rules:

$$\frac{C \vee A \quad C \vee \overline{A}}{C} \quad \frac{C}{C \vee A \quad C \vee \overline{A}}. \quad (3)$$

These rules are called *symmetric cut* and *symmetric weakening*, or *split*, respectively. Note the subtle difference between the symmetric cut rule and the standard cut rule in (1): in the symmetric cut rule, both antecedent formulas have the same *side formula* C . This difference is minor: an application of the non-symmetric cut rule that derives $C \vee D$ from $C \vee A$ and $D \vee \overline{A}$ may be efficiently simulated as follows (here and in what follows, the applicability of the rules has to be read up to associativity, symmetry, and idempotency of disjunctions and conjunctions, and the second consequent of the split rule has been suppressed from the list of derived formulas whenever it is not needed):

1. $C \vee A \vee D$ by split on $C \vee A$,
2. $D \vee \overline{A} \vee C$ by split on $D \vee \overline{A}$,
3. $C \vee D$ by symmetric cut on 1 and 2.

Note also that the rules in (3) do not include a rule for *introduction of conjunction* as in (1). In the presence of the elementary tautologies (or, equivalently, the axiom rule), this difference is again minor: an application of the introduction of conjunction rule that derives $C \vee D \vee (A \wedge B)$ from $C \vee A$ and $D \vee B$ may be efficiently simulated by the following sequence:

1. $\overline{A} \vee \overline{B} \vee (A \wedge B)$ as an elementary tautology,
2. $\overline{A} \vee \overline{B} \vee (A \wedge B) \vee C$ by split on 1,
3. $C \vee A \vee \overline{B} \vee (A \wedge B)$ by split on $C \vee A$,
4. $C \vee \overline{B} \vee (A \wedge B)$ by symmetric cut on 2 and 3,
5. $C \vee \overline{B} \vee (A \wedge B) \vee D$ by split on 4,
6. $D \vee B \vee C \vee (A \wedge B)$ by split on $D \vee B$,
7. $C \vee D \vee (A \wedge B)$ by symmetric cut on 5 and 6.

Thus, for all practical purposes, the Frege proof system as defined in the previous section and the proof system defined here are equivalent. The same observation applies to Resolution. In this case the elementary tautologies are of the form $X \vee \overline{X}$, where X is a variable, and the instances of the symmetric cut and split rules in (3) have a variable for its *principal formula* A . Note that an application of the standard weakening rule that derives the clause $C \vee D$ from the clause C may be efficiently simulated by $|D|$ many applications of the split rule by introducing one literal at a time; here $|D|$ denotes the number of literals in D .

2.5 Sherali-Adams Proof System

Let X_1, \dots, X_n be variables that are intended to range over $\{0, 1\}$, and let $\bar{X}_1, \dots, \bar{X}_n$ be their *twins*, with the intended meaning that $\bar{X}_i = 1 - X_i$. Let A_1, \dots, A_m and A be polynomials on the variables X_1, \dots, X_n and $\bar{X}_1, \dots, \bar{X}_n$. A *Sherali-Adams proof of $A \geq 0$ from $A_1 \geq 0, \dots, A_m \geq 0$* is a polynomial identity of the form

$$\sum_{j=1}^t Q_j P_j = A, \quad (4)$$

where each Q_j is a non-negative linear combination of monomials on the variables X_1, \dots, X_n and $\bar{X}_1, \dots, \bar{X}_n$, and each P_j is a polynomial among A_1, \dots, A_m or one among the following set of *basic* polynomials:

$$\{X_i - X_i^2, X_i^2 - X_i, 1 - X_i - \bar{X}_i, X_i + \bar{X}_i - 1 : i \in [n]\} \cup \{1\}. \quad (5)$$

The *degree* of the proof is the maximum of the degrees of the polynomials $Q_j P_j$ in (4). The *monomial size* of the proof is the sum of the monomial sizes of the polynomials $Q_j P_j$ in (4), where the monomial size of a polynomial is the number of monomials with non-zero coefficient in its unique representation as a linear combination of monomials.

3 Circular Proofs

Informally, a circular proof will be defined as a “proof with cycles”. Formally such objects will be called *circular pre-proofs* because, in general, they are not sound. We define *circular proofs* by adding a global yet efficiently checkable requirement on the definition of pre-proof that guarantees its soundness.

3.1 Circular Pre-Proofs

A *circular pre-proof* is just an ordinary proof with *backedges* that match equal formulas. More formally, a *circular pre-proof* from a set \mathcal{H} of hypothesis formulas is a proof A_1, \dots, A_ℓ from an augmented set of hypothesis formulas $\mathcal{H} \cup \mathcal{B}$, together with a set of *backedges* that is represented by a set $M \subseteq [\ell] \times [\ell]$ of pairs (i, j) , with $j < i$, such that $A_j = A_i$ and $A_j \in \mathcal{B}$. The formulas in the set \mathcal{B} of additional hypotheses are called *bud formulas*.

Just like ordinary proofs are naturally represented by directed acyclic graphs, circular pre-proofs are naturally represented by directed graphs; see Figure 2. For each pair (i, j) in M there is a *backedge* from the formula-vertex of A_i to the formula-vertex of the bud formula A_j ; note that $A_j = A_i$ by definition. By contracting the backedges of a circular pre-proof we get an ordinary directed graph with cycles. If Π is a circular pre-proof, we use $G(\Pi)$ to denote this graph, which we call the *compact graph representation of Π* . Note that $G(\Pi)$ is a bipartite graph with all its edges pointing from a formula-vertex to an inference-vertex, or vice-versa. When Π is clear from the context, we write I and J for the sets of inference- and formula-vertices of $G(\Pi)$, respectively, and $N^-(u)$ and $N^+(u)$ for the sets of in- and out-neighbours of a vertex u of $G(\Pi)$, respectively.

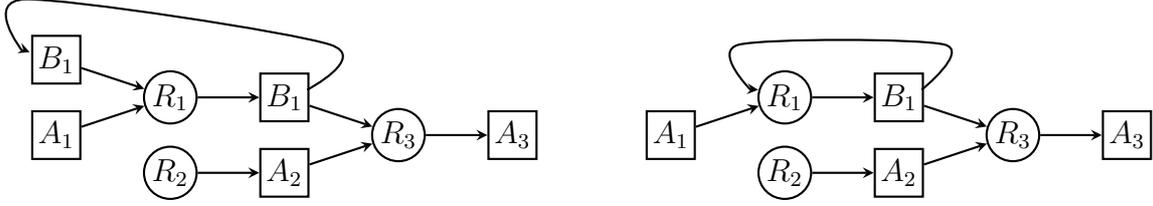


Figure 2: On the left, a circular pre-proof with a unique backedge that links two formula-vertices labelled by the same bud formula B_1 . On the right, a more compact representation of the same circular pre-proof in which the backedge has been contracted.

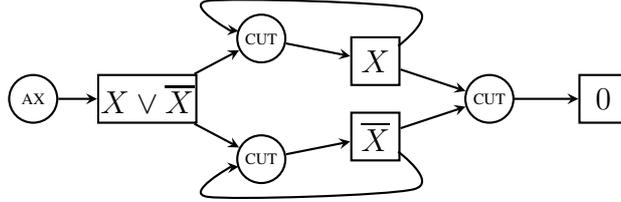


Figure 3: The compact graph representation of an unsound circular pre-proof: the false empty formula 0 is derived from no hypotheses. Note that, no matter what positive weights are assigned to the inference-vertices of the graph, the sum of the weights that enter X minus the sum of the weights that leave X will always be negative (and the same for \bar{X}). As we will see, this turns out to be the *only reason* for it not being sound.

In general, circular pre-proofs need not be sound; see Figure 3 for an example of an unsound circular pre-proof. In order to ensure soundness we need to require a global condition as defined next.

3.2 Circular Proofs

A *flow assignment* for a circular pre-proof Π is an assignment $F : I \rightarrow \mathbb{R}^+$ of positive real weights, or *flows*, where I is the set of inference-vertices of the compact graph representation $G(\Pi)$ of Π . The flow-extended graph that labels each inference-vertex w of $G(\Pi)$ by its flow $F(w)$ is denoted $G(\Pi, F)$. The *inflow* of a formula-vertex in $G(\Pi, F)$ is the sum of the flows of its in-neighbours. Similarly, the *outflow* of a formula-vertex in $G(\Pi, F)$ is the sum of the flows of its out-neighbours. The *balance* of a formula-vertex u of $G(\Pi, F)$ is the inflow minus the outflow of u , and is denoted $B(u)$. In symbols,

$$B(u) := \sum_{w \in N^-(u)} F(w) - \sum_{w \in N^+(u)} F(w). \quad (6)$$

The formula-vertices of strictly negative balance are the *sources* of $G(\Pi, F)$, and those of strictly positive balance are the *sinks* of $G(\Pi, F)$. We think of flow assignments as witnessing a proof of a formula that labels a sink from the set of formulas that label the sources. Concretely, for a given set of hypothesis formulas \mathcal{H} and a given goal formula A , we say that the flow assignment *witnesses a proof of A from \mathcal{H}* if every source of $G(\Pi, F)$ is labelled by a formula in \mathcal{H} , and some sink of $G(\Pi, F)$ is labelled by the formula A .

Finally, a *circular proof of A from \mathcal{H}* is a circular pre-proof for which there exists a flow assignment that witnesses a proof of A from \mathcal{H} . The *length* of a circular proof Π is the number of vertices of $G(\Pi)$, and the *size* of Π is the sum of the sizes of the formulas in the sequence. Note that this definition of size does not depend on the weights that witness the proof. As we will see in the next section, such weights may be assumed to be integral and have small bit-complexity.

3.3 Checking the Global Condition

We still need to argue two facts about circular proofs: 1) that the existence of a witnessing flow assignment guarantees soundness, and 2) that its existence can be checked algorithmically in an efficient way. Soundness is proved in the next section. Here we argue that its existence can be checked efficiently. One way to do this is by solving a linear program.

Lemma 1. *There is a polynomial-time algorithm that, given a circular pre-proof Π , a finite set of hypothesis formulas \mathcal{H} , and a goal formula A as input, returns a flow assignment for Π that witnesses a proof of A from \mathcal{H} , if it exists.*

Proof. Let $V = I \cup J$ be the set of vertices of the compact graph representation $G(\Pi)$ of Π , partitioned into the set I of inference-vertices and the set J of formula-vertices. For each $u \in V$, let $N^-(u)$ and $N^+(u)$ denote the set of in- and out-neighbours of u , respectively. Observe that $N^-(u) \subseteq I$ and $N^+(u) \subseteq J$ for each $u \in J$. Let $H \subseteq J$ be the set of formula-vertices whose labels are in \mathcal{H} , and let $a \in J$ be the formula-vertex whose label is A . For each w in I , let Y_w denote a real-valued variable and consider the following instance of the linear programming feasibility problem:

$$(P) : \begin{array}{ll} \sum_{w \in N^-(u)} Y_w - \sum_{w \in N^+(u)} Y_w \geq 1 & \text{for } u = a, \\ \sum_{w \in N^-(u)} Y_w - \sum_{w \in N^+(u)} Y_w \geq 0 & \text{for each } u \in J \setminus (H \cup \{a\}), \\ Y_w \geq 1 & \text{for each } w \in I. \end{array}$$

We claim that (P) has a feasible solution $(y_v)_{v \in I}$ if and only if there exists a flow assignment $F : I \rightarrow \mathbb{R}^+$ that witnesses a proof of A from \mathcal{H} . For the *only if* direction, define $F : I \rightarrow \mathbb{R}^+$ by $F(w) := y_w$, and read-off the required conditions for F from the inequalities that define (P) . For the *if* direction, define $y_w := F(w)/D$, where D is the minimum in the finite set $\{F(w) : w \in I\} \cup \{B(a)\}$ and $B(a)$ denotes the balance of a defined as in (6). Observe that D is strictly positive by definition. The inequalities of (P) are satisfied by $(y_v)_{v \in I}$ also by definition, and by the choice of D . Since the linear programming feasibility problem can be solved in polynomial time in the size of the input, the lemma follows. \square

By elementary facts of linear programming, it follows from this proof that if there is a flow assignment that witnesses a proof, then there is one with flows that are rational numbers whose bit-complexity is at most polynomial in the length of the circular pre-proof. By taking common denominators and multiplying through, the flows can even be taken to be positive integers of bit-complexity still polynomial in the length of the pre-proof. We collect these observations in a lemma.

Lemma 2. *Let Π be a circular pre-proof of length ℓ . For every flow assignment F for Π there exists another flow assignment F' for Π such that:*

1. for each inference-vertex w of $G(\Pi)$, the flow $F'(w)$ is a positive integer bounded by $\ell!$,
2. the flow-extended graphs $G(\Pi, F)$ and $G(\Pi, F')$ have the same sets of sources and sinks.

Proof. Let I and J be the sets of inference- and formula-vertices of $G(\Pi)$. Let $S \subseteq J$ and $T \subseteq J$ be the sets of sources and sinks of $G(\Pi, F)$, respectively. Consider the following variant of the linear program (P) above:

$$(Q) : \begin{array}{ll} \sum_{w \in N^-(u)} Y_w - \sum_{w \in N^+(u)} Y_w \geq 1 & \text{for each } u \in T, \\ \sum_{w \in N^-(u)} Y_w - \sum_{w \in N^+(u)} Y_w \geq 0 & \text{for each } u \in J \setminus (S \cup T), \\ Y_w \geq 1 & \text{for each } w \in I. \end{array}$$

When we transform (Q) it into standard form by adding exactly $|J| + |I|$ many slack variables, the result will be a linear program of the form $Mx = b$, $x \geq 0$ where x is a vector of $2|I| + |J|$ variables, M is a constraint matrix of dimensions $(|I| + |J|) \times (2|I| + |J|)$, and b is a right-hand side $(|I| + |J|)$ -vector. Moreover, each coefficient in the matrix M and the vector b will be in $\{-1, 0, 1\}$. Since this linear program has a solution (the one given by F adequately extended to the slack variables), it also has a basic feasible solution $(x_u^*)_{u \in V}$. Each component x_u^* is either 0 or, by Cramer's Rule, can be written in the form $\det(N_u) / \det(N)$ where N is a square submatrix of M , and N_u is the matrix that results from replacing the column of N of index u by a subvector of the right-hand side vector b . By ignoring the slack variables we get a solution $(y_w)_{w \in I}$ for (Q) of the same form. Multiplying through by the common denominator $\det(N)$ we get an integral solution $(y'_w)_{w \in I}$ for (Q) whose components have the form $\det(N_w)$; none is 0 because $Y_w \geq 1$ is one of the inequalities in (Q) . Each N_w -matrix has dimensions at most $(|I| + |J|) \times (|I| + |J|)$, and components in $\{-1, 0, 1\}$. It follows that $y'_w = \det(N_w) \leq (|I| + |J|)! = \ell!$. Taking $F'(w) := y'_w$ for each $w \in I$ completes the proof. \square

3.4 Soundness of Circular Proofs

In this section we develop the soundness proof when the set \mathcal{R} of inference rules is fixed to axiom, symmetric cut, and split. See Section 2 for a discussion on this choice of rules. In the next section we discuss the general case.

We give two different proofs: one combinatorial and one (semi-)algebraic.

Theorem 3. *Let \mathcal{R} be the set of inference rules made of axiom, symmetric cut, and split. Let \mathcal{H} be a set of hypothesis formulas and let A be a goal formula. If there is a circular proof of A from \mathcal{H} through the rules in \mathcal{R} , then every truth assignment that satisfies every formula in \mathcal{H} also satisfies A .*

First proof. Fix a truth assignment α . We prove the stronger claim that, for every circular pre-proof Π from an unspecified set of hypothesis formulas, every integral flow assignment F for Π , and every sink s of $G(\Pi, F)$, if α falsifies the formula that labels s , then α also falsifies the formula that labels some source of $G(\Pi, F)$. The restriction to integral flow assignments is no loss of generality by Lemma 2, and allows a proof by induction on the total flow-sum of F ; the sum of the flows assigned by F .

If the total flow-sum is zero, then there are no sinks and the statement holds vacuously. Assume then that the total flow-sum is positive, and let s be a sink of $G(\Pi, F)$, with balance $B(s) > 0$, whose labelling formula B is falsified by α . Since its balance is positive, s must

have at least one in-neighbour r . Since the consequent formula of the rule at r is falsified by α , some antecedent formula of the rule at r must exist that is also falsified by α . Let u be the corresponding in-neighbour of r , and let $B(u)$ be its balance. If $B(u)$ is negative, then u is a source of $G(\Pi, F)$, and we are done. Assume then that $B(u)$ is non-negative.

Let $\delta := \min\{B(s), F(r)\}$ and note that $\delta > 0$ because $B(s) > 0$ and $F(r) > 0$. We define a new circular pre-proof Π' and an integral flow assignment F' for Π' to which we will apply the induction hypothesis. The construction will guarantee the following properties:

1. $G(\Pi')$ is a subgraph of $G(\Pi)$ with the same set of formula-vertices,
2. the total flow-sum of F' is smaller than the total flow-sum of F .
3. u is a sink of $G(\Pi', F')$ and s is not a source of $G(\Pi', F')$,
4. if t is a source of $G(\Pi', F')$, then t is a source of $G(\Pi, F)$ or an out-neighbour of r in $G(\Pi)$.

From this the claim will follow by applying the induction hypothesis to Π' , F' and u . Indeed the induction hypothesis applies to them by Properties 1, 2 and the first half of 3, and it will give a source t of $G(\Pi', F')$ whose labelling formula is falsified by α . We argue that t must also be a source of $G(\Pi, F)$, in which case we are done. To argue for this, assume otherwise and apply Property 4 to conclude that t is an out-neighbour of r in $G(\Pi)$, which by the second half of Property 3 must be different from s because t is a source of $G(\Pi', F')$. Recall now that s is a second out-neighbour of r . This can be the case only if r is a split inference, in which case the formulas that label s and t must be of the form $C \vee B$ and $C \vee \overline{B}$, respectively, for appropriate formulas C and B . But, by assumption, α falsifies the formula that labels s , namely $C \vee B$, which means that α satisfies the formula $C \vee \overline{B}$ that labels t . This is the contradiction we were after.

It remains to construct Π' and F' that satisfy properties 1, 2, 3, and 4. We define them by cases according to whether $F(r) > B(s)$ or $F(r) \leq B(s)$, and then argue for the correctness of the construction. In case $F(r) > B(s)$, and hence $\delta = B(s)$, let Π' be defined as Π without change, and let F' be defined by $F'(r) := F(r) - \delta$ and $F'(w) := F(w)$ for every other $w \in I \setminus \{r\}$. Obviously Π' is still a valid pre-proof and F' is a valid flow assignment for Π' by the assumption that $F(r) > B(s) = \delta$. In case $F(r) \leq B(s)$, and hence $\delta = F(r)$, let Π' be defined as Π with the inference-step that labels r removed, and let F' be defined by $F'(w) := F(w)$ for every $w \in I \setminus \{r\}$. Note that in this case Π' is still a valid pre-proof but perhaps from a larger set of hypothesis formulas.

In both cases the proof of the claim that Π' and F' satisfy Properties 1, 2, 3, and 4 is the same. Property 1 is obvious in both cases. Property 2 follows from the fact that the total flow-sum of F' is the total flow-sum of F minus δ , and $\delta > 0$. The first half of Property 3 follows from the fact that the balance of u in $G(\Pi', F')$ is $B(u) + \delta$, while $B(u) \geq 0$ by assumption and $\delta > 0$. The second half of Property 3 follows from the fact that the balance of s in $G(\Pi', F')$ is $B(s) - \delta$, while $B(s) \geq \delta$ by choice of δ . Property 4 follows from the fact that the only formula-vertices of $G(\Pi', F')$ of balance smaller than that in $G(\Pi, F)$ are the out-neighbours of r . This completes the proof of the claim, and of the theorem. \square

We give a second very different proof of soundness that will play an important role later.

Second proof. Let Π be a circular pre-proof and let F be a flow assignment for Π that witnesses a proof of A from \mathcal{H} . Let α be a truth assignment that satisfies all the formulas in \mathcal{H} , and let s

be an arbitrary formula-vertex in $G(\Pi)$. We show that if α falsifies the formula that labels s , then s is not a sink of $G(\Pi, F)$.

Let $V = I \cup J$ be the set of vertices of $G(\Pi)$ partitioned into the set I of inference-vertices, and the set J of formula-vertices. For every $u \in J$, let A_u be the formula that labels u and let $Z_u := \alpha(A_u)$; the truth-value that α gives to A_u . By inspection of the three allowed inference rules, for each $w \in I$ with labelling inference rule R and in- and out-neighbours N^- and N^+ , respectively, we have:

$$\begin{aligned} -(1 - Z_a) &\geq 0 && \text{if } R = \text{axiom with } N^+ = \{a\}, \\ (1 - Z_a) + (1 - Z_b) - (1 - Z_c) &\geq 0 && \text{if } R = \text{cut with } N^- = \{a, b\} \text{ and } N^+ = \{c\}, \\ (1 - Z_a) - (1 - Z_b) - (1 - Z_c) &\geq 0 && \text{if } R = \text{split with } N^- = \{a\} \text{ and } N^+ = \{b, c\}. \end{aligned}$$

Multiplying each such inequality by the positive flow $F(w)$ of w and adding up over all $w \in I$ we get

$$\sum_{w \in I} F(w) \left(\sum_{v \in N^-(w)} (1 - Z_v) - \sum_{v \in N^+(w)} (1 - Z_v) \right) \geq 0 \quad (7)$$

Rearranging the sum by formula-vertices, as opposed to by inference-vertices, we get

$$\sum_{u \in J} (1 - Z_u) \left(\sum_{w \in N^+(u)} F(w) - \sum_{w \in N^-(u)} F(w) \right) = - \sum_{u \in J} B(u)(1 - Z_u) \geq 0, \quad (8)$$

where $B(u)$ is the balance of u in $G(\Pi, F)$. Now, $Z_u = 1$ whenever u is a source, $Z_s = 0$ for s by assumption, and $B(u)(1 - Z_u) \geq 0$ for every other formula-vertex $u \in I$ by the definition of circular proof. Hence

$$-B(s) \geq 0, \quad (9)$$

which shows that s has non-positive balance in $G(\Pi, F)$ and is thus not a sink. \square

3.5 Soundness for Other Sets of Rules

We claim that both proofs of soundness that we gave apply without change to any set of sound inference rules that have a single consequent formula. This requirement is fulfilled by all sets of standard inference rules, such as (1), and is subsumed by the following more general but technical one:

(*) Any inference rule in \mathcal{R} that has more than one consequent formula has the property that any truth assignment that falsifies one of its consequent formulas must satisfy all other consequent formulas.

Obviously, if all rules in \mathcal{R} have a single consequent, then (*) is satisfied. Note also that the only rule that has more than one consequent formula among axiom, symmetric cut, and split is split, and clearly it has the required property. Thus, the following statement generalizes Theorem 3.

Theorem 4. *Let \mathcal{R} be a set of sound inference rules that satisfy property (*). Let \mathcal{H} be a set of hypothesis formulas and let A be a goal formula. If there is a circular proof of A from \mathcal{H} through the rules in \mathcal{R} , then every truth assignment that satisfies every formula in \mathcal{H} also satisfies A .*

Proof. The first proof of Theorem 3 was already phrased in a way that the generalization to sets of inference rules that satisfy (*) is straightforward. We discuss the generalization of the second proof. Let Π be a circular proof with rules in \mathcal{R} , let A_u be the formula that labels the formula-vertex u , let $Z_u := \alpha(A_u)$ be the truth value given to A_u by a truth assignment α , and let w be an inference-vertex of Π with in- and out-neighbors N^- and N^+ , respectively. Then, the following inequality holds:

$$\sum_{u \in N^-} (1 - Z_u) - \sum_{u \in N^+} (1 - Z_u) \geq 0 \quad (10)$$

Indeed, if $Z_a = 0$ for some $a \in N^+$, then by the soundness of the rule there exists $b \in N^-$ such that $Z_b = 0$, and by (*) we also have $Z_c = 1$ for every $c \in N^+ \setminus \{a\}$. The conclusion to this is that the left summand in (10) is at least 1, and the right summand in (10) is at most 1, so their difference is non-negative. From here it suffices to note that this is the only property we used in order to derive equations (7), (8) and (9). \square

4 Circular Frege vs Tree-Like Frege

For some weak proof systems, such as Resolution, it makes a great deal of difference whether the proof-graph has tree-like structure or not [6]. For stronger proofs systems, such as Frege, this is not the case. Indeed Tree-like Frege polynomially simulates Dag-like Frege, and this holds true of any inference-based proof system with the set of all formulas as its set of allowed formulas, and a finite set of inference rules that is implicationally complete [16]. Since circular proofs further generalize the structure of the proof-graph, it is interesting to discuss whether circular proofs in Frege are complexity-wise more powerful than standard Frege proofs.

It turns out that this is not the case. In this section we show how to efficiently simulate Circular Frege, as defined in Section 3, by standard Frege proofs.

Theorem 5. *Tree-like Frege and Circular Frege polynomially simulate each other.*

The main idea underlying the simulation of Circular Frege by standard Frege is to formalize, in standard Dag-like Frege itself, the LP-based proof of soundness of Frege circular proofs; cf. the *second proof* of Theorem 3. To do that we use a formalization of linear arithmetic in Frege, due to Buss [9] and Goerdt [13], which was originally designed to simulate counting arguments and Cutting Planes in Frege. Since Cutting Planes subsumes LP-reasoning, the core of the LP-based proof of Theorem 3 can be formalized in it.

4.1 Formalization of Linear Arithmetic in Frege

We collect the relevant parts of Goerdt's results in a single theorem, but before that we need to introduce some notation. Let \mathcal{L}_n denote the collection of all linear inequalities of the form

$$a_1 X_1 + \cdots + a_n X_n \geq b, \quad (11)$$

where X_1, \dots, X_n are formal variables and a_1, \dots, a_n and b are integers. Note that there is a natural inclusion embedding of \mathcal{L}_n into \mathcal{L}_{n+1} by padding each inequality with $a_{n+1} = 0$. Accordingly, by a slight abuse of the notation, we think of ℓ as in (11) as an inequality of \mathcal{L}_m

for each $m \geq n$, and we define $\ell \equiv \ell'$ to hold if ℓ and ℓ' are two inequalities from \mathcal{L} that represent the same inequality up to padding by zero coefficients.

Let \mathcal{L} be the set of all linear inequalities with integer coefficients; i.e., the union of all the \mathcal{L}_n 's. Let ℓ and ℓ' denote two inequalities in \mathcal{L} , with sequences of coefficients b, a_1, \dots, a_n and b', a'_1, \dots, a'_n , respectively, when they are padded by zeros to the smallest common length $n + 1$. Let c be a positive integer. We write $c \cdot \ell$ and $\ell + \ell'$ for the following two inequalities, respectively:

$$\begin{aligned} ca_1X_1 + \dots + ca_nX_n &\geq cb, \\ (a_1 + a'_1)X_1 + \dots + (a_n + a'_n)X_n &\geq (b + b'). \end{aligned}$$

Let \mathcal{F} denote the collection of all propositional formulas in negation normal form. We move back and forth between the truth assignments and the 0-1 assignments for the same sets of variables through the natural correspondance that identifies 1 with *true* and 0 with *false*. If $f : \{X_1, \dots, X_n\} \rightarrow \{0, 1\}$ denotes such an assignment, and ℓ and A denote an inequality and a formula on the variables X_1, \dots, X_n , then we write $f(\ell)$ and $f(A)$ for their truth values under f . Concretely, if ℓ is as in (11), then $f(\ell)$ is true if and only if $a_1f(X_1) + \dots + a_nf(X_n)$ is at least b .

For the purposes of computability, an inequality ℓ as in (11) is represented by the sequence of the binary encodings of its coefficients b, a_1, \dots, a_n , and has size

$$(n + 1) + \log_2(|b|) + \sum_{i=1}^n \log_2(|a_i|), \quad (12)$$

with the convention that $\log_2(0) = 0$.

Theorem 6 ([13]). *There is a mapping $I : \mathcal{L} \rightarrow \mathcal{F}$ that takes linear inequalities to formulas and that has the following properties. For every two inequalities ℓ and ℓ' in \mathcal{L} , every positive integer c , every truth assignment f , and every variable X , the following hold:*

1. $I(\ell)$ has size polynomial in the size of ℓ ,
2. $I(\ell)$ has the same variables as ℓ , and $f(\ell) = f(I(\ell))$,
3. there is a polynomial-size Frege proof of $I(\ell)$ from $I(\ell')$ whenever $\ell \equiv \ell'$,
4. there is a polynomial-size Frege proof of $I(\ell + \ell')$ from $I(\ell)$ and $I(\ell')$,
5. there is a polynomial-size Frege proof of $I(c \cdot \ell)$ from $I(\ell)$,
6. there is a polynomial-size Frege proof of $I(\ell)$ from $I(c \cdot \ell)$,
7. there is a polynomial-size Frege proof of $I(X \geq 1)$ from X ,
8. there is a polynomial-size Frege proof of $I(-X \geq 0)$ from \overline{X} ,
9. there is a polynomial-size Frege proof of $I(-X \geq -1)$ from nothing,
10. there is a polynomial-size Frege proof of $I(X \geq 0)$ from nothing,
11. there is a polynomial-size Frege proof of $I(0 \geq 0)$ from nothing,
12. there is a polynomial-size Frege proof of 0 from $I(0 \geq 1)$.

Moreover, the mapping I and the Frege proofs in 3–12 are all computable in time that is bounded by a fixed polynomial in the sizes of the input and the output inequalities.

Proof. All this can be found in Goerdts's article [13], which in turn builds on Buss's seminal [9]: the definition of the mapping I is in Section 2.6 of Goerdts's article, Properties 1–3, and 7–12 follow by inspection of the definition of I given there, and Properties 4, 5, and 6 are Theorems 3.1, 3.5, and 3.6 in Goerdts's article, respectively. \square

4.2 Proof of the Simulation

This section will be devoted to the proof of Theorem 4. The statement that Circular Frege polynomially simulates Tree-like Frege follows from the discussion in Section 2.4. We concentrate on the reverse simulation. Since it is known that Tree-like Frege polynomially simulates Dag-like Frege, it suffices to do the simulation through dag-like proofs. Also we claim that it suffices to do the simulation only for refutations. Indeed, from a short circular proof of A from \mathcal{H} we can get a short circular refutation of $\mathcal{H} \cup \{\overline{A}\}$ by adding a cut between the derived A and the new hypothesis \overline{A} , with flow equal to the balance of the formula-vertex of A . And from a short dag-like refutation of $\mathcal{H} \cup \{\overline{A}\}$ we can get a short dag-like proof of A from \mathcal{H} by replacing each use of the hypothesis formula \overline{A} by the axiom instance $A \vee \overline{A}$.

Let Π be a Circular Frege refutation of a set of hypothesis formulas \mathcal{H} . The simulation goes in three steps. In the first step we build a linear program $P = \{\ell_1, \dots, \ell_m\}$ that has one variable Z_u for each formula-vertex of $G(\Pi)$, whose infeasibility witnesses the soundness of Π as a circular refutation. This is done by closely following the second proof of soundness of circular proofs; cf. Theorem 3. In the second step we apply Theorem 6 to convert an LP-based infeasibility witness for P into a Frege refutation of the set of formulas $\mathcal{H}' := \{I(\ell_1), \dots, I(\ell_m)\}$. Here I is the mapping from Theorem 6. In the third step we apply the substitution defined by $Z_u := A_u$ to this Frege refutation, where A_u is the formula that labels the formula-vertex u , and we apply Theorem 6 again in order to show that each formula in the substituted \mathcal{H}' has an efficient Frege proof from \mathcal{H} .

First step. The linear program P has one variable Z_u for each formula-vertex $u \in J$ in $G(\Pi)$, and two sets of inequalities $P_J = \{\ell_u : u \in J\}$ and $P_I = \{\ell_w : w \in I\}$ indexed by the sets of formula-vertices J and inference-vertices I of $G(\Pi)$, respectively. Concretely, for each formula-vertex $u \in J$ the inequality ℓ_u is defined as follows:

$$\begin{aligned} -Z_u &\geq 0 && \text{if } u \text{ is the formula-vertex of the derived empty formula,} \\ -(1 - Z_u) &\geq 0 && \text{if } u \text{ is a formula-vertex of a hypothesis formula,} \\ (1 - Z_u) &\geq 0 && \text{if } u \text{ is any other formula-vertex.} \end{aligned}$$

For each inference-vertex w with labelling inference rule R and in- and out-neighbours N^- and N^+ , respectively, the inequality ℓ_w is defined as follows:

$$\begin{aligned} -(1 - Z_a) &\geq 0 && \text{if } R = \text{axiom with } N^+ = \{a\}, \\ (1 - Z_a) + (1 - Z_b) - (1 - Z_c) &\geq 0 && \text{if } R = \text{cut with } N^- = \{a, b\} \text{ and } N^+ = \{c\}, \\ (1 - Z_a) - (1 - Z_b) - (1 - Z_c) &\geq 0 && \text{if } R = \text{split with } N^- = \{a\} \text{ and } N^+ = \{b, c\}. \end{aligned}$$

A certificate of the infeasibility of $P = P_I \cup P_J$ is given by two assignments of non-negative weights $(b_u : u \in J)$ and $(c_w : w \in I)$ for the inequalities in P_J and P_I , respectively, in such a way that the corresponding positive linear combination

$$\sum_{u \in J} b_u \cdot \ell_u + \sum_{w \in I} c_w \cdot \ell_w \tag{13}$$

simplifies to the trivially false inequality $0 \geq 1$. In turn, such an assignment of weights can be shown to exist from the assumption that Π is a valid circular refutation: let F be a flow assignment that witnesses that Π is a valid proof, let s be the formula-vertex of the derived empty formula, and set $b_u := -B(u)/B(s)$ for each formula-vertex $u \in J$ that is a source of $G(\Pi, F)$, set $b_u := B(u)/B(s)$ for each formula-vertex $u \in J$ that is not a source of $G(\Pi, F)$, and set $c_w := F(w)/B(s)$ for every inference-vertex $w \in I$, where $B(u)$ denotes the balance of $u \in J$ in $G(\Pi, F)$. Note that $B(s)$ is strictly positive because s must be a sink of $G(\Pi, F)$. This means that each b_u is well-defined and non-negative because the balance of all formula-vertices except the sources is non-negative in $G(\Pi, F)$. The proof that this assignment of weights satisfies the requirement that (13) simplifies to $0 \geq 1$ is precisely the content of the second proof of Theorem 3. This completes the first step of the simulation.

Second step. The second step is a direct application of Theorem 6: Define the non-negative integers $b'_u = b_u \cdot B(s)$ and $c'_w = c_w \cdot B(s)$. Start at $\mathcal{H}' = \{I(\ell_u) : u \in J\} \cup \{I(\ell_w) : w \in I\}$. By 5 and 11, obtain Frege proofs of $I(b'_u \cdot \ell_u)$ and $I(c'_w \cdot \ell_w)$ for each $u \in J$ and each $w \in I$. Now let ℓ' denote the positive linear combination defined as in (13) with b_u and c_w replaced by b'_u and c'_w , respectively. Recall that ℓ is $-1 \geq 0$ and hence ℓ' is $-B(s) \geq 0$. By 4, obtain Frege proofs of $I(-B(s) \geq 0)$. By 6, obtain Frege proofs of $I(-1 \geq 0)$, and finally, by 3 and 12, obtain the Frege proofs of $I(0 \geq 1)$ and 0, respectively.

Third step. We start the third step by applying the substitution defined by $Z_u := A_u$ to the refutation of \mathcal{H}' , where A_u is again the formula that labels the formula-vertex u . For each $v \in I \cup J$, let $I(\ell_v)^*$ denote the result of applying this substitution to $I(\ell_v)$. To complete the step we need to get polynomial-size Frege proofs of $I(\ell_v)^*$ from \mathcal{H} , for each $v \in I \cup J$. We do this as a less direct application of Theorem 6.

For each formula-vertex $u \in J$ of a hypothesis formula in \mathcal{H} , we get a Frege proof of $I(\ell_u)^*$ from A_u by applying the substitution $X := A_u$ to the Frege proof given by 7 in Theorem 6. When u is the formula-vertex of the derived empty formula, we get a Frege proof of $I(\ell_u)^*$ from $\bar{0}$ by applying the substitution $X := 0$ to the Frege proof given by 8 in Theorem 6. Since $\bar{0}$ is the consequent of an instance of the axiom rule of Frege (namely $0 \vee \bar{0}$), this is a Frege proof of $I(\ell_u)^*$ from nothing. For every other formula-vertex u , we get a Frege proof of $I(\ell_u)^*$ from nothing by applying the substitution $X := A_u$ to the Frege proof given by 9 in Theorem 6.

For each inference-vertex $w \in I$, with labelling rule R and in- and out-neighbours N^- and N^+ , we proceed as follows. By 2 in Theorem 6 and the soundness of R , first note that $I(\ell_w)^*$ is a propositional tautology. We claim that, in addition, this tautology is obtained by applying a substitution to another tautology T that has at most two propositional variables X and Y . Concretely, T will itself be the result of applying a substitution to $I(\ell_w)$. We define T by cases depending on what rule R is. If R is the axiom rule and $N^+ = \{a\}$, then we take T to be the result of applying the substitution $Z_a := X \vee \bar{X}$ to $I(\ell_w)$. If R is the cut rule, $N^- = \{a, b\}$ and $N^+ = \{c\}$, then we take T to be the result of applying the substitution $Z_a := Y \vee X$, $Z_b := Y \vee \bar{X}$, and $Z_c := Y$ to $I(\ell_w)$. If R is the split rule, $N^- = \{a\}$ and $N^+ = \{b, c\}$, then we take T to be the result of applying the substitution $Z_a := Y$, $Z_b := Y \vee X$, and $Z_c := Y \vee \bar{X}$ to $I(\ell_w)$. By 2 in Theorem 6 and the soundness of R , in all three cases T is a tautology with at most two propositional variables. By the completeness of Frege, T has a constant-size Frege proof. Applying the substitution that turns T into $I(\ell_w)^*$ to this proof we get a polynomial-size Frege proof of $I(\ell_w)^*$ as desired. This completes the third step, and the proof.

5 Circular Resolution

In this section we investigate the power of Circular Resolution. Recall from the discussion in Section 2.4 that Resolution is traditionally defined to have cut as its only rule, but that an essentially equivalent version of it is obtained if we define it through symmetric cut, split, and axiom, still all restricted to clauses. This more liberal definition of Resolution, while staying equivalent vis-a-vis the tree-like and dag-like versions of Resolution, will play an important role for the circular version of Resolution.

While for Frege proof systems we proved that there is no qualitative difference between tree-like, dag-like, and circular proofs, in this section we show that circular Resolution can be exponentially stronger than dag-like Resolution. Indeed, we show that Circular Resolution is polynomially equivalent with the Sherali-Adams proof system, which is already known to be stronger than dag-like Resolution:

Theorem 7. *Sherali-Adams and Circular Resolution polynomially simulate each other. Moreover, the simulation one way converts degree into width (exactly), and the simulation in the reverse way converts width into degree (also exactly).*

For the statement of Theorem 7 to even make sense, Sherali-Adams is to be understood as a proof system for deriving clauses from clauses, under an appropriate encoding of clauses.

5.1 Pigeonhole Principles

Let G be a bipartite graph with vertex bipartition (U, V) , and set of edges $E \subseteq U \times V$. For a vertex $w \in U \cup V$, we write $N_G(w)$ to denote the set of neighbours of w in G , and $\deg_G(w)$ to denote its degree. The Graph Pigeonhole Principle of G , denoted G -PHP, is a CNF formula that has one variable $X_{u,v}$ for each edge (u, v) in E and the following set clauses:

$$\begin{array}{ll} \frac{X_{u,v_1} \vee \cdots \vee X_{u,v_d}}{X_{u_1,v} \vee X_{u_2,v}} & \begin{array}{l} \text{for } u \in U \text{ with } N_G(u) = \{v_1, \dots, v_d\}, \\ \text{for } u_1, u_2 \in U \text{ and } v \in V \text{ with } u_1 \neq u_2, \text{ and } v \in N_G(u_1) \cap N_G(u_2). \end{array} \end{array}$$

If $|U| > |V|$, and in particular if $|U| = n + 1$ and $|V| = n$, then G -PHP is unsatisfiable by the pigeonhole principle. For $G = K_{n+1,n}$, the complete bipartite graph with sides of sizes $n + 1$ and n , the formula G -PHP is the standard CNF encoding PHP_n^{n+1} of the pigeonhole principle.

Even for certain constant degree bipartite graphs with $|U| = n + 1$ and $|V| = n$, the formulas are hard for Resolution.

Theorem 8 ([5, 15]). *There are families of bipartite graphs $(G_n)_{n \geq 1}$, where G_n has maximum degree bounded by a constant and vertex bipartition (U, V) of G_n that satisfies $|U| = n + 1$ and $|V| = n$, such that every Resolution refutation of G_n -PHP has width $\Omega(n)$ and length $2^{\Omega(n)}$. Moreover, this implies that every Resolution refutation of PHP_n^{n+1} has length $2^{\Omega(n)}$.*

In contrast, we show that these formulas have Circular Resolution refutations of polynomial length and, simultaneously, constant width.

Theorem 9. *For every bipartite graph G of maximum degree d with bipartition (U, V) such that $|U| > |V|$, there is a Circular Resolution refutation of G -PHP of length polynomial in $|U| + |V|$ and width d . In particular, PHP_n^{n+1} has a Circular Resolution refutation of polynomial length.*

Proof. We are going to build the refutation in pieces. Concretely, for every $u \in U$ and $v \in V$, we describe a Circular Resolution proofs $\Pi_{u \rightarrow}$ and $\Pi_{\rightarrow v}$, with their associated flow assignments. These proofs will have width bounded by $\deg_G(u)$ and $\deg_G(v)$, respectively, and size polynomial in $\deg_G(u)$ and $\deg_G(v)$, respectively. Moreover, the following properties will be ensured:

1. The proof-graph of $\Pi_{u \rightarrow}$ contains a formula-vertex labelled by the empty clause 0 with balance +1 and a formula-vertex labelled $\overline{X_{u,v}}$ with balance -1 for every $v \in N_G(u)$; any other formula-vertex that has negative balance is labelled by a clause of G -PHP.
2. The proof-graph $\Pi_{\rightarrow v}$ contains a formula-vertex labelled by the empty clause 0 with balance -1 and a formula-vertex labelled by $\overline{X_{u,v}}$ with balance +1 for every $u \in N_G(v)$; any other formula-vertex that has negative balance is labelled by a clause of G -PHP.

By patching these pieces together through the formula-vertices that have the same label we get a proof in which all the formula-vertices that have negative balance are clauses of G -PHP, and the empty clause 0 has balance $|U| - |V| > 0$. This is indeed a Circular Resolution refutation of G -PHP.

For the construction of $\Pi_{u \rightarrow}$, rename the neighbours of u as $1, 2, \dots, \ell$. Let C_j denote the clause $X_{u,1} \vee \dots \vee X_{u,j}$ and note that C_ℓ is a clause of G -PHP. Split $\overline{X_{u,\ell}}$ on $X_{u,1}$, then on $X_{u,2}$, and so on up to $X_{u,\ell-1}$ until we produce $\overline{X_{u,\ell}} \vee C_{\ell-1}$. Then resolve this clause with C_ℓ to produce $C_{\ell-1}$. The same construction starting at $\overline{X_{u,\ell-1}}$ and $C_{\ell-1}$ produces $C_{\ell-2}$. Repeating ℓ times we get down to the empty clause.

For the construction of $\Pi_{\rightarrow v}$ we need some more work. Again rename the neighbours of v as $1, 2, \dots, \ell$. We define a sequence of proofs Π_1, \dots, Π_ℓ inductively. The base case Π_1 is just one application of the split rule to the empty clause to derive $X_{1,v}$ and $\overline{X_{1,v}}$, with flow 1. Proof Π_{i+1} is built using Π_i as a component. Let $X_{i+1,v} \vee \Pi_i$ denote the proof that is obtained from adding the literal $X_{i+1,v}$ to every clause in Π_i . First we observe that $X_{i+1,v} \vee \Pi_i$ has balance -1 on $X_{i+1,v}$ and balance +1 on, among other clauses, $X_{i+1,v} \vee \overline{X_{j,v}}$ for $j = 1, \dots, i$. Each such clause can be resolved with clause $\overline{X_{i+1,v}} \vee \overline{X_{j,v}}$ to produce the desired clauses $\overline{X_{j,v}}$ with balance +1. Splitting the empty clause on variable $X_{i+1,v}$ would even out the balance of the formula-vertex labelled by $X_{i+1,v}$ and produce a vertex labelled by $\overline{X_{i+1,v}}$ of balance +1. We take the final Π_ℓ as $\Pi_{\rightarrow v}$. \square

5.2 Simulation by Sherali-Adams

In this section we prove one half of Theorem 7. We need some preparation. Fix a set of variables X_1, \dots, X_n and their twins $\overline{X}_1, \dots, \overline{X}_n$. For a clause $C = \bigvee_{j \in Y} X_j \vee \bigvee_{j \in Z} \overline{X}_j$ with $Y \cap Z = \emptyset$, define

$$T(C) := - \prod_{j \in Y} \overline{X}_j \prod_{j \in Z} X_j, \quad (14)$$

Observe that a truth assignment satisfies C if and only if the corresponding 0-1 assignment for the variables of $T(C)$ makes the inequality $T(C) \geq 0$ true. There is an alternative encoding of clauses into inequalities that is sometimes used. Define

$$L(C) := \sum_{j \in Y} X_j + \sum_{j \in Z} \overline{X}_j - 1, \quad (15)$$

and observe that a truth assignment satisfies C if and only if the corresponding 0-1 assignment makes the inequality $L(C) \geq 0$ true. We state the results of this section for the M -encoding of

clauses, but the same result would hold for the L -encoding because there is efficient SA proof of (14) from (15) (see Lemma 4.2 in [3]), and vice-versa.

We will use the following lemma, which is a variant of Lemma 4.4 in [3]:

Lemma 10. *Let C be a clause with w literals and let X be a variable that does not appear in C . Then the following four inequalities have Sherali-Adams proofs (from nothing) of constant monomial size and degree $w + 1$:*

1. $T(X \vee \bar{X}) \geq 0$,
2. $-T(C \vee \bar{X}) - T(C \vee X) + T(C) \geq 0$,
3. $-T(C) + T(C \vee \bar{X}) + T(C \vee X) \geq 0$,
4. $-T(C) \geq 0$.

Proof. This is straightforward. Let $C = \bigvee_{i \in Y} X_i \vee \bigvee_{j \in Z} X_j$. Then

1. $T(X \vee \bar{X}) = (1 - X - \bar{X}) \cdot X + (X^2 - X)$,
2. $-T(C \vee \bar{X}) - T(C \vee X) + T(C) = (X + \bar{X} - 1) \cdot \prod_{i \in Y} \bar{X}_i \prod_{j \in Z} X_j$,
3. $-T(C) + T(C \vee \bar{X}) + T(C \vee X) = (1 - X - \bar{X}) \cdot \prod_{i \in Y} \bar{X}_i \prod_{j \in Z} X_j$,
4. $-T(C) = 1 \cdot \prod_{i \in Y} \bar{X}_i \prod_{j \in Z} X_j$.

The claim on the monomial size and the degree follows. □

Now we are ready to state and prove the first half of Theorem 7.

Lemma 11. *Let A_1, \dots, A_m and A be non-tautological clauses. If there is a Circular Resolution proof of A from A_1, \dots, A_m of length s and width w , then there is a Sherali-Adams proof of $T(C) \geq 0$ from $T(A_1) \geq 0, \dots, T(A_m) \geq 0$ of monomial size $3s$ and degree w .*

Proof. Let Π be a Circular Resolution proof of A from A_1, \dots, A_m , and let F be the corresponding flow assignment. Let I and J be the sets of inference- and formula-vertices of $G(\Pi)$, and let $B(u)$ denote the balance of formula-vertex $u \in J$ in $G(\Pi, F)$. For each formula-vertex $u \in J$ labelled by formula A_u , define the polynomial $P_u := T(A_u)$. For each inference-vertex $w \in I$ labelled by rule R , with sets of in- and out-neighbours N^- and N^+ , respectively, define the polynomial

$$\begin{aligned} P_w &:= T(A_a) && \text{if } R = \text{axiom with } N^+ = \{a\}, \\ P_w &:= -T(A_a) - T(A_b) + T(A_c) && \text{if } R = \text{cut with } N^- = \{a, b\} \text{ and } N^+ = \{c\}, \\ P_w &:= -T(A_a) + T(A_b) + T(A_c) && \text{if } R = \text{split with } N^- = \{a\} \text{ and } N^+ = \{b, c\}. \end{aligned}$$

By double counting, the following polynomial identity holds:

$$\sum_{u \in J} B(u) P_u = \sum_{w \in I} F(w) P_w. \tag{16}$$

Let s be the sink of $G(\Pi, F)$ that is labelled by the derived clause A . Since $B(s) > 0$, equation (16) rewrites into

$$\sum_{w \in I} \frac{F(w)}{B(s)} P_w + \sum_{u \in J \setminus \{s\}} -\frac{B(u)}{B(s)} P_u = P_s.$$

We claim that this identity is a legitimate Sherali-Adams proof of $T(A) \geq 0$ from the inequalities $T(A_1) \geq 0, \dots, T(A_m) \geq 0$. First, $P_s = T(A_s) = T(A)$, i.e. the right-hand side is correct. Second, each term $(F(w)/B(s))P_w$ for $w \in I$ is a sum of legitimate terms of a Sherali-Adams proof by the definition of P_w and Parts 1, 2 and 3 of Lemma 10. Third, since each source $u \in I$ of $G(\Pi, F)$ has $B(u) < 0$ and is labelled by a formula in A_1, \dots, A_m , the term $(-B(u)/B(s))P_u$ of a source $u \in I$ is a positive multiple of $T(A_u)$ and hence also a legitimate term of a Sherali-Adams proof from $T(A_1) \geq 0, \dots, T(A_m) \geq 0$. And forth, since each non-source $u \in I$ of $G(\Pi, F)$ has $B(u) \geq 0$, each term $(-B(u)/B(s))P_u$ of a non-source $u \in I$ is a sum of legitimate terms of a Sherali-Adams proof by the definition of P_u and Part 4 of Lemma 10. The monomial size and degree of this Sherali-Adams proof are as claimed, and the proof of the Lemma is complete. \square

5.3 Simulation of Sherali-Adams

In this section we prove the other half of Theorem 7. We use the notation from Section 5.2.

Lemma 12. *Let A_1, \dots, A_m and A be non-tautological clauses. If there is a Sherali-Adams proof of $T(A) \geq 0$ from $T(A_1) \geq 0, \dots, T(A_m) \geq 0$ of monomial size s and degree d , then there is a Circular Resolution proof of A from A_1, \dots, A_m of length $O(s)$ and width d .*

Proof. Fix a Sherali-Adams proof of $T(A) \geq 0$ from $T(A_1) \geq 0, \dots, T(A_m) \geq 0$, say

$$\sum_{j=1}^t Q_j P_j = T(A), \quad (17)$$

where each Q_j is a non-negative linear combination of monomials on the variables X_1, \dots, X_n and $\bar{X}_1, \dots, \bar{X}_n$, and each P_j is a polynomial from among $T(A_1), \dots, T(A_m)$ or from among the polynomials in the list (5) from the definition of Sherali-Adams in Section 2.

Our goal is to massage the proof (17) until it becomes a Circular Resolution proof in disguise. Towards this, as a first step, we claim that (17) can be transformed into a *normalized proof* of the form

$$\sum_{j=1}^{t'} Q'_j P'_j = T(A) \quad (18)$$

that has the following properties:

1. each Q'_j is a positive multiple of a multilinear monomial, and $Q'_j P'_j$ is multilinear,
2. each P'_j is a polynomial among $T(A_1), \dots, T(A_m)$, or among the polynomials in the set

$$\{-X_i \bar{X}_i, 1 - X_i - \bar{X}_i, X_i + \bar{X}_i - 1 : i \in [n]\} \cup \{1\}. \quad (19)$$

Comparing (19) with the original list (5) in the definition of Sherali-Adams, note that we have replaced the polynomials $X_i - X_i^2$ and $X_i^2 - X_i$ by $-X_i \bar{X}_i$. Note also that, by splitting the Q_j 's into their terms, we may assume without loss of generality that each Q_j in (17) is a positive multiple of a monomial on the variables X_1, \dots, X_n and $\bar{X}_1, \dots, \bar{X}_n$.

In order to prove the claim we rely on the well-known fact that each real-valued function over Boolean domain has a unique representation as a multilinear polynomial:

Fact 13. For every natural number N and every function $f : \{0, 1\}^N \rightarrow \mathbb{R}$ there is a unique multilinear polynomial P with N variables satisfying $P(a_1, \dots, a_N) = f(a_1, \dots, a_N)$ for every $a_1, \dots, a_N \in \{0, 1\}$.

With this fact in hand, it suffices to convert each $Q_j P_j$ in the left-hand side of (17) into a $Q'_j P'_j$ of the required form (or 0), and check that $Q_j P_j$ and $Q'_j P'_j$ are equivalent over the 0-1 assignments to its variables (without relying on the constraint that $\bar{X}_i = 1 - X_i$). The claim will follow from the combination of Fact 13 and the fact that $T(A)$ is multilinear since, by assumption, A is non-tautological.

We proceed to the conversion of each $Q_j P_j$ into a $Q'_j P'_j$ of the required form. Recall that we assumed already, without loss of generality, that each Q_j is a positive multiple of a monomial. The multilinearization of a monomial Q_j is the monomial $M(Q_j)$ that results from replacing every factor Y^k with $k \geq 2$ in Q_j by Y . Obviously Q_j and $M(Q_j)$ agree on 0-1 assignments, but replacing each Q_j by $M(Q_j)$ is not enough to guarantee the normal form that we are after. We need to proceed by cases on P_j .

If P_j is one of the polynomials among $T(A_1), \dots, T(A_m)$, say $T(A_i)$, then we let Q'_j be $M(Q_j)$ with every variable that appears in A_i deleted, and let P'_j be $T(A_i)$ itself. It is obvious that this works. If P_j is $1 - X_i - \bar{X}_i$, then we proceed by cases on whether Q_j contains X_i or \bar{X}_i or both. If Q_j contains neither X_i nor \bar{X}_i , then the choice $Q'_j = M(Q_j)$ and $P'_j = P_j$ works. If Q_j contains X_i or \bar{X}_i , call it Y , but not both, then the choice $Q'_j = M(Q_j)/Y$ and $P'_j = -X_i \bar{X}_i$ works. If Q_j contains both X_i and \bar{X}_i , then the choice $Q'_j = M(Q_j)/(X_i \bar{X}_i)$ and $P'_j = -X_i \bar{X}_i$ works. If P_j is $X_i + \bar{X}_i - 1$, then again we proceed by cases on whether Q_j contains X_i or \bar{X}_i or both. If Q_j contains neither X_i nor \bar{X}_i , then the choice $Q'_j = M(Q_j)$ and $P'_j = P_j$ works. If Q_j contains X_i or \bar{X}_i , call it Y , but not both, then the choice $Q'_j = M(Q_j)Y$ and $P'_j = 1$ works. If Q_j contains both X_i and \bar{X}_i , then the choice $Q'_j = M(Q_j)$ and $P'_j = 1$ works. If P_j is the polynomial 1, then the choice $Q'_j = M(Q_j)$ and $P'_j = 1$ works. Finally, if P_j is of the form $X_i - X_i^2$ or $X_i^2 - X_i$, then we replace $Q_j P_j$ by 0. Observe that in this case $Q_j P_j$ is always 0 over 0-1 assignments, and the conversion is correct. This completes the proof that (18) exists.

It remains to be seen that the normalized proof (18) is a Circular Resolution proof in disguise. For each $j \in [m]$, let a_j and M_j be the positive real and the multilinear monomial, respectively, such that $Q'_j = c_j \cdot M_j$. Let also C_j be the unique clause on the variables X_1, \dots, X_n such that $T(C_j) = -M_j$. Let $[t']$ be partitioned into five sets $I_0 \cup I_1 \cup I_2 \cup I_3 \cup I_4$ where

1. I_0 is the set of $j \in [t']$ such that $P'_j = T(A_{i_j})$ for some $i_j \in [m]$,
2. I_1 is the set of $j \in [t']$ such that $P'_j = -X_{i_j} \bar{X}_{i_j}$ for some $i_j \in [n]$,
3. I_2 is the set of $j \in [t']$ such that $P'_j = 1 - X_{i_j} - \bar{X}_{i_j}$ for some $i_j \in [n]$,
4. I_3 is the set of $j \in [t']$ such that $P'_j = X_{i_j} + \bar{X}_{i_j} - 1$ for some $i_j \in [n]$,
5. I_4 is the set of $j \in [t']$ such that $P'_j = 1$.

Define new polynomials P''_j as follows:

$$\begin{aligned} P''_j &:= T(C_j \vee A_{i_j}) \text{ for } j \in I_0, \\ P''_j &:= T(C_j \vee \bar{X}_{i_j} \vee X_{i_j}) \text{ for } j \in I_1, \\ P''_j &:= -T(C_j) + T(C_j \vee \bar{X}_{i_j}) + T(C_j \vee X_{i_j}) \text{ for } j \in I_2, \\ P''_j &:= -T(C_j \vee \bar{X}_{i_j}) - T(C_j \vee X_{i_j}) + T(C_j) \text{ for } j \in I_3, \end{aligned}$$

$P_j'' := T(C_j)$ for $j \in I_4$.

With this notation, (18) rewrites into

$$\sum_{j \in I_0} a_j P_j'' + \sum_{j \in I_1} a_j P_j'' + \sum_{j \in I_2} a_j P_j'' + \sum_{j \in I_3} a_j P_j'' = T(A) + \sum_{j \in I_4} a_j P_j''. \quad (20)$$

Finally we are ready to construct the circular proof. We build it by listing the inference-vertices with their associated flows, and then we identify together all the formula-vertices that are labelled by the same clause.

Intuitively, I_0 's are weakenings of hypothesis clauses, I_1 's are weakenings of axioms, I_2 's are cuts, and I_3 's are splits. Formally, each $j \in I_0$ becomes a chain of $|C_j|$ many split vertices that starts at the hypothesis clause A_{i_j} and produces its weakening $C_j \vee A_{i_j}$; all split vertices in this chain have flow a_j . Each $j \in I_1$ becomes a sequence that starts at one axiom vertex that produces $X_{i_j} \vee \overline{X_{i_j}}$ with flow a_j , followed by a chain of $|C_j|$ many split vertices that produces its weakening $C_j \vee X_{i_j} \vee \overline{X_{i_j}}$; all split vertices in this chain also have flow a_j . Each $j \in I_2$ becomes one cut vertex that produces C_j from $C_j \vee X_{i_j}$ and $C_j \vee \overline{X_{i_j}}$ with flow a_j . And each $j \in I_3$ becomes one split vertex that produces $C_j \vee X_{i_j}$ and $C_j \vee \overline{X_{i_j}}$ from C_j with flow a_j .

This defines the inference-vertices of the proof graph. The construction is completed by introducing one formula-vertex for each different clause that is an antecedent or a consequent of these inference-vertices. The construction was designed in such a way that equation (20) is the proof that, in this proof graph and its associated flow assignment, the following hold:

1. there is a sink with balance 1 and that is labelled by A ,
2. for each $j \in I_0$ there is a source with balance $-a_j$ that is labelled by A_{i_j} ,
3. all other formula-vertices have non-negative balance.

This proves that the construction is a correct Circular Resolution proof of A from A_1, \dots, A_m . The claim that the length of this proof is $O(s)$ and its width is d follows by inspection. \square

6 Concluding Remarks

One interesting and immediate consequence of the degree/width equivalence between Sherali-Adams and Circular Resolution, as stated in Theorem 7, is that there is a polynomial-time algorithm that automates the search for Circular Resolution proofs of bounded width:

Corollary 14. *There is an algorithm that, given an integer parameter w and a set of clauses A_1, \dots, A_m and A with n variables, returns a width- w Circular Resolution proof of A from A_1, \dots, A_m , if there is one, and the algorithm runs in time polynomial in m and n^w .*

The proof-search algorithm of Corollary 14 relies on linear programming because it relies on our translations to and from Sherali-Adams, whose automating algorithm does rely on linear programming. A direct proof of Corollary 14 is, however, also possible: Lay down a formula-vertex for each clause of width at most w . Add axiom inference-vertices for all axioms in the list. Connect triples of such clauses through appropriate cut or split inference-vertices; if one clause follows by cut from the other two clauses in the set, connect them through a cut vertex, and if two clauses follow by split from the other clause in the set, connect them through a split vertex.

Finally run the algorithm that finds an appropriate flow assignment if it exists, cf. Lemma 1. Of course this proof-search algorithm is also based on linear programming, and it remains as an open problem whether a more combinatorial algorithm exists for the same task. Could perhaps a matching-based algorithm exist?

Yet another consequence of the equivalence between Circular Resolution and Sherali-Adams is that Circular Resolution has a length-width relationship in the style of the one due to Ben-Sasson and Wigderson for Dag-like Resolution [5]. This follows from Theorem 7 in combination with the size-degree relationship that is known to hold for Sherali-Adams (see [20, 2]). As a consequence to this, exponential length lower bounds follow from linear width lower bounds for Circular Resolution, or equivalently, from linear degree lower bounds for Sherali-Adams. In particular, since linear degree lower bounds for Sherali-Adams are known for 3-CNF formulas with expanding incidence graphs (see [14, 20]), we get the following:

Corollary 15. *There are families of 3-CNF formulas $(F_n)_{n \geq 1}$, where F_n has $O(n)$ variables and $O(n)$ clauses, such that every Circular Resolution refutation of F_n has width $\Omega(n)$ and size $2^{\Omega(n)}$.*

It should be noticed that, unlike the well-known observation that tree-like and dag-like width are equivalent measures for Resolution, dag-like and circular width are *not* equivalent for Resolution. The sparse graph pigeonhole principle from Section 5.1 illustrates the point. This shows that bounded-width circular Resolution proofs cannot be *unfolded* into bounded-width tree-like Resolution proofs in any natural (except infinitary?) way.

This observation also explains, perhaps, why our proof that Circular Frege simulates Tree-like Frege goes via a very indirect translation, and raises one further question (and answer). It is known that Tree-like Bounded-Depth Frege simulates Dag-like Bounded-Depth Frege, at the cost of increasing the depth by one. Could the simulation of Circular Frege by Tree-like Frege be made to preserve bounded depth? The (negative) answer is also provided by the pigeonhole principle which is known to be hard for Bounded-Depth Frege [1, 19, 17], but is easy for Circular Resolution, and hence for Circular Depth-1 Frege.

One last aspect of the equivalence between Circular Resolution and Sherali-Adams concerns the theory of SAT-solving. As is well-known, state-of-the-art SAT-solvers produce Resolution proofs as certificates of unsatisfiability and, as a result, will not be able to handle counting arguments of pigeonhole type. This has motivated the study of so-called *pseudo-Boolean solvers* that handle counting constraints and reasoning through specialized syntax and inference rules. The equivalence of Circular Resolution and Sherali-Adams suggests a completely different approach to incorporate counting capabilities: instead of enhancing the syntax, keep it to clauses but enhance the *proof-shapes*. Whether circular proof-shapes can be handled in a sufficiently effective and efficient way is of course in doubt, but certainly a question worth studying.

Acknowledgments Both authors were partially funded by European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme, grant agreement ERC-2014-CoG 648276 (AUTAR). First author partially funded by MINECO through TIN2013-48031-C4-1-P (TASSAT2). We acknowledge the work of Jordi Coll who conducted experimental results for finding and visualizing actual circular resolution proofs of small instances of the sparse pigeonhole principle.

References

- [1] Miklos Ajtai. The complexity of the pigeonhole principle. In *29th Annual Symposium on Foundations of Computer Science*, pages 346–355, Oct 1988.
- [2] Albert Atserias. Selected topics on semi-algebraic proof complexity. Slides at Workshop on Proof Complexity and Beyond, Oberwolfach Research Institute for Mathematics, August 2017. Available at author’s webpage.
- [3] Albert Atserias, Massimo Lauria, and Jakob Nordström. Narrow proofs may be maximally long. *ACM Trans. Comput. Log.*, 17(3):19:1–19:30, 2016.
- [4] Yu Hin Au and Levent Tunçel. A comprehensive analysis of polyhedral lift-and-project methods. *SIAM J. Discrete Math.*, 30(1):411—451, 2016.
- [5] Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow - resolution made simple. *J. ACM*, 48(2):149–169, 2001.
- [6] Maria Luisa Bonet, Juan Luis Esteban, Nicola Galesi, and Jan Johannsen. On the relative complexity of resolution refinements and cutting planes proof systems. *SIAM J. Comput.*, 30(5):1462–1484, 2000.
- [7] James Brotherston. *Sequent Calculus Proof Systems for Inductive Definitions*. PhD thesis, University of Edinburgh, November 2006.
- [8] James Brotherston and Alex Simpson. Sequent calculi for induction and infinite descent. *Journal of Logic and Computation*, 21(6):1177–1216, 2011.
- [9] Samuel R. Buss. Polynomial size proofs of the propositional pigeonhole principle. *Journal of Symbolic Logic*, 52(4):916–927, 1987.
- [10] Stefan S. Dantchev. Rank complexity gap for lovász-schrijver and sherali-adams proof systems. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 311–317, 2007.
- [11] Stefan S. Dantchev, Barnaby Martin, and Mark Nicholas Charles Rhodes. Tight rank lower bounds for the sherali-adams proof system. *Theor. Comput. Sci.*, 410(21-23):2054–2063, 2009.
- [12] Christian Dax, Martin Hofmann, and Martin Lange. *A Proof System for the Linear Time μ -Calculus*, pages 273–284. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.
- [13] Andreas Goerdt. Cutting plane versus frege proof systems. In Egon Börger, Hans Kleine Büning, Michael M. Richter, and Wolfgang Schönfeld, editors, *Computer Science Logic: 4th Workshop, CSL ’90 Heidelberg, Germany, October 1–5, 1990 Proceedings*, pages 174–194, Berlin, Heidelberg, 1991. Springer Berlin Heidelberg.
- [14] Dima Grigoriev. Linear lower bound on degrees of positivstellensatz calculus proofs for the parity. *Theoretical Computer Science*, 259(1):613 – 622, 2001.

- [15] Armin Haken. The intractability of resolution. *Theoretical Computer Science*, 39:297 – 308, 1985. Third Conference on Foundations of Software Technology and Theoretical Computer Science.
- [16] Jan Krajíček. *Bounded Arithmetic, Propositional Logic, and Complexity Theory*. Cambridge, 1994.
- [17] Jan Krajíček, Pavel Pudlák, and Alan Woods. An exponential lower bound to the size of bounded depth frege proofs of the pigeonhole principle. *Random Structures and Algorithms*, 7(1):15–39, 1995.
- [18] Damian Niwiński and Igor Walukiewicz. Games for the μ -calculus. *Theoretical Computer Science*, 163(1):99 – 116, 1996.
- [19] Toniann Pitassi, Paul Beame, and Russell Impagliazzo. Exponential lower bounds for the pigeonhole principle. *Computational Complexity*, 3:97–140, 1993.
- [20] Toniann Pitassi and Nathan Segerlind. Exponential lower bounds and integrality gaps for tree-like lovász-schrijver procedures. *SIAM J. Comput.*, 41(1):128–159, 2012.
- [21] Hanif D. Sherali and Warren P. Adams. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM J. Discrete Math.*, 3(3):411—430, 1990.
- [22] Thomas Studer. On the proof theory of the modal mu-calculus. *Studia Logica*, 89(3):343–363, Aug 2008.