Focussing, MALL and the polynomial hierarchy

Anupam Das

University of Copenhagen

February 20, 2018

Abstract. We investigate how to extract alternating time bounds from 'focussed' proofs, treating non-invertible rule phases as nondeterministic computation and invertible rule phases as co-nondeterministic computation. We refine the usual presentation of focussing to account for deterministic computations in proof search, which correspond to invertible rules that do not branch, more faithfully associating phases of focussed proof search to their alternating time complexity.

As our main result, we give a focussed system for *affine* MALL (i.e. with weakening) and give encodings to and from true quantified Boolean formulas (QBFs): in one direction we encode QBF satisfiability and in the other we encode focussed proof search. Moreover we show that the composition of the two encodings preserves quantifier alternation, hence yielding natural fragments of affine MALL complete for each level of the polynomial hierarchy. This refines the well-known result that affine MALL is **PSPACE**-complete.

1 Introduction and motivation

Proof systems are often a source of optimal decision algorithms for logics, theoretically speaking. We now know how to extract various bounds for proof search as a function of certain properties of the proof system at hand. For instance we may compute:

- nondeterministic time bounds via *proof complexity*, e.g. [6,11,5];
- (non)deterministic space bounds via the *depth* of proofs or search spaces, and *loop-checking*, e.g. [2,10,21];
- deterministic or co-nondeterministic time bounds via systems of *invertible rules*, see e.g. [25,19].

However, despite considerable progress in the field, there still remains a gap between the obtention of (co-)nondeterministic time bounds, such as **NP** or **coNP**, and space bounds such as **PSPACE** (equivalently, *alternating polynomial time*, cf. [3]). Phrased differently, while we have many logics we know to be **PSPACE**-complete (intuitionistic logic, various modal logics, etc.), we have very little understanding of their fragments corresponding to subclasses of **PSPACE**. In particular, in this work we are interested in the levels of the *polynomial hierarchy* (**PH**) [24], which correspond to alternating polynomial-time Turing machines with *boundedly* many alternations.

One relevant development in structural proof theory in the last 20-30 years has been the notion of *focussing*, e.g. [1,12,14]. Focussed systems elegantly delineate the phases of invertible and non-invertible inferences in proofs, allowing the natural obtention of alternating time bounds for a logic. Furthermore, they significantly constrain the number of local choices available, resulting in reduced nondeterminism during proof search, while remaining complete (the 'focussing theorem'). Such systems thus serve as a natural starting point for identifying fragments of **PSPACE**-complete logics complete for levels of **PH**.

In this work we will consider the case of *multiplicative additive linear logic* (MALL) [9], often seen as the prototypical system for **PSPACE** since its proof rules constitute the abstract templates of terminating proof search. (Indeed, MALL is well-known to be **PSPACE**-complete [15,16].) By considering a focussed presentation of the *affine* variant MALLw, which admits weakening, we analyse proof search to identify classes of theorems belonging to each level of **PH**.[1] To demonstrate the accuracy of this method, we also show that these classes are, in fact, *complete* for their respective levels, via encodings from true quantified Boolean formulas (QBFs) of appropriate quantifier complexity, cf. [3].

One shortfall of focussed systems is that, in their usual form, they unfortunately do not make adequate consideration for *deterministic* computations, which correspond to invertible rules that do not branch, and so the natural measure of complexity there ('decide depth') can considerably overestimate the alternating complexity of a theorem. In the worst case this can lead to rather degenerate bounds, exemplified in [7] where an encoding of SAT in intuitionistic logic requires a linear decide depth, despite being **NP**-complete.[2] In this work we keep the same abstract notion of focussing, but split the usual invertible, or 'asynchronous', phase into a 'deterministic' phase, with non-branching invertible rules, and a 'co-nondeterministic' phase, with branching invertible rules. In this way, when expressing proof search as an alternating predicate, a $\forall$ quantifier needs only be introduced in a co-nondeterministic phase. It turns out that this adaptation suffices to obtain the tight bounds we are after.

This paper is structured as follows. In Sect. 2 we present preliminaries on QBFs and alternating time complexity, and in Sect. 3 we present preliminaries on MALL and focussing. In Sect. 4 we present an encoding of true QBFs into MALLw, tracking the association between quantifier complexity and 'decide depth' in focussed proof search. In Sect. 5 we briefly explain how provability predicates for focussed systems may be obtained as QBFs, with quantifier complexity calibrated appropriately with decide depth (the 'focussing hierarchy'). In Sect. 6 we show how this depth measure can be feasibly approximated to yield a bona fide encoding of MALLw back into true QBFs. Furthermore, we show that the composition of the two encodings preserves quantifier complexity, and yields fragments of MALLw complete for each level of the polynomial hierarchy. Finally, in Sect. 7 we give some concluding remarks regarding the case of (non-affine) MALL, and further perspectives on our presentation of focussing.

[1] MALLw is also **PSPACE**-complete, a folklore result subsumed by this work.

[2] In fact the same phenomenon presents in this work, cf. Fig. 3.

2 Preliminaries on logic and computational complexity

We will recall some basic theory of Boolean logic, and its connections to alternating time complexity. Throughout this paper we omit constants (or 'units'), both for classical and linear logic, to simplify exposition and avoid clashing notations.

2.1 Second-order Boolean logic

Quantified Boolean formulas (QBFs) are obtained from the language of classical propositional logic by adding (second-order) quantifiers varying over propositions. Formally, let us fix some set Var of propositional *variables*, written x, y etc. QBFs, written φ, ψ etc., are generated as follows:

$$\varphi \quad ::= \quad x \mid \overline{x} \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x.\varphi \mid \forall x.\varphi$$

The formula $\overline{x}$ stands for the *negation* of x, and all formulas we deal with will be in *De Morgan normal form*, i.e. with negation restricted to variables as in the grammar above. Nonetheless, we may sometimes write $\overline{\varphi}$ to denote the De Morgan *dual* of φ, generated by the following identities:

$$\overline{\overline{\varphi}} := \varphi \qquad \begin{array}{rcl} \overline{(\varphi \vee \psi)} & := & \overline{\varphi} \wedge \overline{\psi} \\ \overline{(\varphi \wedge \psi)} & := & \overline{\varphi} \vee \overline{\psi} \end{array} \qquad \begin{array}{rcl} \overline{\exists x.\varphi} & := & \forall x.\overline{\varphi} \\ \overline{\forall x.\varphi} & := & \exists x.\overline{\varphi} \end{array}$$

A formula is *closed* if all its variables are bound by a quantifier ($\exists$ or $\forall$). We write $|\varphi|$ for the number of literal occurrences in φ.

An *assignment* is a subset $\alpha \subseteq \mathsf{Var}$. We define the *satisfaction* relation between an assignment α and a formula φ, written $\alpha \vDash \varphi$, in the usual way:

- $\alpha \vDash x$ if $x \in \alpha$.
- $\alpha \vDash \overline{x}$ if $x \notin \alpha$.
- $\alpha \vDash \varphi \vee \psi$ if $\alpha \vDash \varphi$ or $\alpha \vDash \psi$.
- $\alpha \vDash \varphi \wedge \psi$ if $\alpha \vDash \varphi$ and $\alpha \vDash \psi$.
- $\alpha \vDash \exists x.\varphi$ if $\alpha \setminus \{x\} \vDash \varphi$ or $\alpha \cup \{x\} \vDash \varphi$.
- $\alpha \vDash \forall x.\varphi$ if $\alpha \setminus \{x\} \vDash \varphi$ and $\alpha \cup \{x\} \vDash \varphi$.

Definition 1 (Quantified Boolean logic). *A QBF φ is* satisfiable *if there is some assignment $\alpha \subseteq \mathsf{Var}$ such that $\alpha \vDash \varphi$. It is* valid *if $\alpha \vDash \varphi$ for every assignment $\alpha \subseteq \mathsf{Var}$. If φ is closed, then we may simply say that it is* true, *written $\vDash \varphi$, when it is satisfiable and/or valid.*[3]

Second-order classical propositional logic *(*$\mathsf{CPL2}$*) is the set of true QBFs.*

In practice, when dealing with a given formula φ, we will only need to consider assignments α that contain variables occurring in φ. We will assume this later when we discuss predicates (or 'languages') computed by open QBFs.

[3] Notice that, by definition of satisfaction these two notions coincide for closed QBFs.

We point out that, from the logical point of view, it suffices to work with only closed QBFs, with satisfiability recovered by prenexing $\exists$ quantifiers and validity recovered by prenexing $\forall$ quantifiers; in the presence of units/constants, the definition of 'truth' above could be adapted with no reference to α. However we will also make use of open formulas in this work to describe languages/predicates, so it will be useful to have the notion of satisfaction available.

Definition 2 (QBF hierarchy). *For $k \geq 0$ we define the following classes:*

- $\Sigma_0^q = \Pi_0^q$ *is the set of quantifier-free QBFs.*
- $\Sigma_{k+1}^q \supseteq \Pi_k^q$ *and, if $\varphi \in \Sigma_{k+1}^q$, then so is $\exists x.\varphi$.*
- $\Pi_{k+1}^q \supseteq \Sigma_n^q$ *and, if $\varphi \in \Pi_{k+1}^q$, then so is $\forall x.\varphi$.*

Notice that we have only defined the classes above for 'prenexed' QBFs, i.e. with all quantifiers at the front. It is well known that any QBF is equivalent to such a formula. For this reason we will systematically assume that all QBFs we deal with are in prenex form. Notice that $\varphi \in \Sigma_k^q$ if and only if $\overline{\varphi} \in \Pi_k^q$, by the definition of De Morgan duality.

In this work we will not need to formally deal with any deduction system for $\mathsf{CPL2}$, although we point out that there is a simple system, *semantic trees*, whose proof search dynamics closely match quantifier complexity [13].

2.2 Alternating time complexity

In computation we are used to the distinction between *deterministic* and *nondeterministic* computation. Intuitively, *co-nondeterminism* is just the 'dual' of nondeterminism: at the machine level it is captured by 'nondeterministic' Turing machines where *every* run is accepting, not just *some* run as in the case of usual nondeterminism. From here *alternating* Turing machines generalise both the nondeterministic and co-nondeterministic models by allowing both universally branching states and existentially branching states.

Intuitions aside, we will introduce the concepts we need here assuming only a familiarity with deterministic and nondeterministic Turing machines and their complexity measures, to limit the formal prerequisites. Our exposition is informal, but the reader may find comprehensive details in, e.g., [22].

For a language L, we write $\mathbf{NP}(L)$ to mean the class of languages accepted in polynomial time by some nondeterministic Turing machine which may, at any point, query in constant time whether some word is in L or not. We extend this to classes of languages $\mathcal{C}$, writing $\mathbf{NP}(\mathcal{C})$ for $\bigcup\limits_{L \in \mathcal{C}} \mathbf{NP}(L)$. We also write $\mathbf{co}\mathcal{C}$ for the class of languages whose complements are in $\mathcal{C}$.

Definition 3 (Polynomial hierarchy, [24]). *We define the following classes:*

- $\Sigma_0^p = \Pi_0^p := \mathbf{P}$.
- $\Sigma_{k+1}^p := \mathbf{NP}(\Sigma_k^p)$.
- $\Pi_{k+1}^p := \mathbf{co}\Sigma_{k+1}^p$.

The polynomial hierarchy (**PH**) *is* $\bigcup_{k\geq 0} \Sigma^p_k = \bigcup_{k\geq 0} \Pi^p_k$.

We may more naturally view the polynomial hierarchy as the bounded-quantifier-alternation fragments of QBFs we introduced earlier. For this we construe Σ^q_k and Π^q_k as classes of finite *languages*, by associating with a QBF $\varphi(\vec{x})$ the class of (finite) assignments $\alpha \subseteq \vec{x}$ satisfying it. (Assignments themselves may be seen as binary strings of length $|\vec{x}|$ which encode their characteristic functions.) Σ^q_k*-satisfaction* (Π^q_k*-satisfaction*) is the problem of deciding, given a Σ^q_k (resp. Π^q_k) formula $\varphi(\vec{x})$ and an assignment $\alpha \subseteq \vec{x}$, whether $\alpha \vDash \varphi(\vec{x})$.

Theorem 4 (cf. [3]). *For $k \geq 1$, the Σ^q_k-satisfaction (Π^q_k-satisfaction) is Σ^p_k-complete (resp. Π^p_k-complete).*

Corollary 5 (of Thm. 4). *For $k \geq 1$, the class of true closed Σ^q_k QBFs is Σ^p_k-complete, and the class of true closed Π^q_k QBFs is Π^p_k-complete.*

3 Linear logic and proof search

In this section we introduce *multiplicative additive linear logic* (MALL) and its proof theory [9], in particular a certain *focussed* proof system for it, cf. [1,8,4].

3.1 Multiplicative additive linear logic

For convenience, we will work with the same set Var of variables that we used for QBFs and, as for classical logic, we will omit constants/units for simplicity. To distinguish them from QBFs, we will use the metavariables A, B, etc. for MALL formulas, generated as follows:

$$A \quad ::= \quad x \mid \overline{x} \mid A \parr B \mid A \oplus B \mid A \otimes B \mid A \,\&\, B$$

$\parr, \otimes$ are called *multiplicative* connectives, and $\oplus, \&$ are called *additive* connectives. Like for QBFs, we have restricted negation to the variables, thanks to De Morgan duality in MALL. Again we may write $\overline{A}$ for the De Morgan dual of A, which is generated similarly to the case of QBFs:

$$\overline{\overline{A}} \;:=\; A \qquad \begin{array}{rcl} \overline{(A \parr B)} & := & \overline{A} \otimes \overline{B} \\ \overline{(A \otimes B)} & := & \overline{A} \parr \overline{B} \end{array} \qquad \begin{array}{rcl} \overline{(A \oplus B)} & := & \overline{A} \,\&\, \overline{B} \\ \overline{(A \,\&\, B)} & := & \overline{A} \oplus \overline{B} \end{array}$$

Due to De Morgan duality, we will work only with 'one-sided' calculi for MALL, where all formulas occur to the right of the sequent arrow. This means we will have fewer cases to consider for formal proofs, although we will also informally adopt a two-sided notation when it is convenient.

Definition 6 (Proof system). *A* cedent*, written Γ, Δ etc., is just a multiset of formulas; as usual we use the comma, ',', for multiset union. The system (cut-free)* MALL *is given in Fig. 1. The system* MALLw*, also called* affine MALL*, is defined in the same way, only with the id rule replaced by:*

$$\mathit{wid}\,\frac{}{\vdash \Gamma, x, \overline{x}} \tag{1}$$

$$
{\scriptstyle id}\,\frac{}{\vdash x, \overline{x}} \quad {\scriptstyle \parr}\,\frac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B} \quad {\scriptstyle \otimes}\,\frac{\vdash \Gamma, A \quad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \otimes B} \quad {\scriptstyle \oplus}\,\frac{\Gamma, A_i}{\Gamma, A_0 \oplus A_1}\, i \in \{0,1\} \quad {\scriptstyle \&}\,\frac{\vdash \Gamma, A \quad \vdash \Gamma, B}{\vdash \Gamma, A \,\&\, B}
$$

Fig. 1. The system (cut-free) MALL.

Notice that, following the tradition in linear logic, we write '$\vdash$' for the sequent arrow, though we point out that the deduction theorem does not actually hold w.r.t. linear implication. For the affine variant, we have simply built weakening into the identity step, since it may always be permuted upwards in a proof:

Proposition 7 (Weakening admissibility). *The following rule, called* weakening*, is (height-preserving) admissible in* MALLw*:*

$$
{\scriptstyle wk}\,\frac{\vdash \Gamma}{\vdash \Gamma, A}
$$

Notice also that we have not included the 'cut' rule, thanks to cut-elimination for linear logic [9]. It will play no role in this paper.

3.2 (Multi-)focussed systems for proof search

Focussed systems for MALL (and linear logic in general) have been widely studied [1,12,8,4]. The idea is to associate polarities to the connectives based on whether their rule is invertible (negative) or their dual's rule is invertible (positive). Now bottom-up proof search can be organised in a manner where, once we have chosen a non-invertible principal formula to decompose (the 'focus'), we may continue to decompose its auxiliary formulas until the focus becomes invertible. The main result herein is the *completeness* of such proof search strategies, known as the *focussing theorem* (a.k.a. the 'focalisation theorem').

It is known that 'multi-focussed' variants, where one may have many foci, can lead to 'canonical' representations of proofs for MALL [4]. Furthermore, the alternation behaviour of focussed proof search can be understood via a game theoretic approach [8]. However, such frameworks unfortunately fall short of characterising the alternating complexity of proof search in a faithful way. The issue is that the usual focussing methodology does not make any account for *deterministic* computations, which correspond to invertible rules that do not branch. Such rules are usually treated just like the other invertible rules, and so morally introduce extraneous quantifiers when encoding proof search as an alternating time predicate.

For these reasons we introduce a bespoke presentation of (multi-)focussing for MALL, with a designated 'deterministic' phase allowing invertible non-branching rules, in this case the $\parr$ rule. To avoid conflicts with more traditional presentations, we refer to the other two phases as 'nondeterministic' and 'co-nondeterministic' rather than 'synchronous' and 'asynchronous' respectively; at the same time this reinforces the intended connections to computational complexity.

Deterministic phase:

$$\text{id}\,\frac{}{\vdash a, \overline{a}} \qquad \text{⅋}\,\frac{\vdash \Gamma, A, B}{\vdash \Gamma, A ⅋ B} \qquad D\,\frac{\vdash \vec{a}, \vec{P} \Downarrow \vec{P}'}{\vdash \vec{a}, \vec{P}, \vec{P}'}\ \vec{P}' \neq \varnothing \qquad \bar{D}\,\frac{\vdash \vec{a}, \vec{P} \Uparrow \vec{M}}{\vdash \vec{a}, \vec{P}, \vec{M}}\ \vec{M} \neq \varnothing$$

Nondeterministic phase:

$$\oplus\,\frac{\vdash \Gamma \Downarrow \Delta, A_i}{\vdash \Gamma \Downarrow \Delta, A_0 \oplus A_1}\ i \in \{0,1\} \qquad \otimes\,\frac{\vdash \Gamma \Downarrow \Sigma, A \quad \Delta \Downarrow \Pi, B}{\vdash \Gamma, \Delta \Downarrow \Sigma, \Pi, A \otimes B} \qquad R\,\frac{\vdash \Gamma, \vec{a}, \vec{N}}{\vdash \Gamma \Downarrow \vec{a}, \vec{N}}$$

Co-nondeterministic phase:

$$\&\,\frac{\vdash \Gamma \Uparrow \Delta, A \quad \Gamma \Uparrow \Delta, B}{\vdash \Gamma \Uparrow \Delta, A \& B} \qquad \bar{R}\,\frac{\vdash \Gamma, \vec{P}, \vec{O}}{\vdash \Gamma \Uparrow \vec{P}, \vec{O}}$$

Fig. 2. The system FMALL.

In what follows, we use a, b, etc. to vary over literals. We also use the following metavariables to vary over formulas with the corresponding top-level connectives:

M : 'negative and not deterministic'	$\&$
N : 'negative'	$\&, ⅋$
O : 'deterministic'	$\otimes, ⅋, a$
P : 'positive'	$\otimes, \oplus$
Q : 'positive and not deterministic'	$\oplus$

'Vectors' are used to vary over multisets of associated formulas, e.g. $\vec{P}$ varies over multisets of P-formulas. Sequents may now contain the delimiters $\Downarrow$ or $\Uparrow$.

Definition 8 (Multi-focussed proof system). *We define the (multi-focussed) system* FMALL *in Fig. 2. The system* FMALLw *is the same as* FMALL *but with the id rule redefined as* (1)*.*

Note that the determinism of $\otimes$ plays no role in this one-sided calculus, but in a two-sided calculus we would have a full symmetry of rules. A proof of a formula A is simply a proof of the sequent $\vdash A$, i.e. there is no need to pre-decorate with arrows thanks to the deterministic phase. The rules D and $\bar{D}$ are called *decide* and *co-decide* respectively, while R and $\bar{R}$ are called *release* and *co-release* respectively. We have not included a 'store' rule, for simplicity, but if we did we would also recover a dual 'co-store' rule.

As usual for multi-focussed systems, the analogous focussed system can be recovered by restricting to only one focussed formula in a nondeterministic phase. However, in our presentation, we may also impose the dual reststriction, that there is only one formula in 'co-focus' during a co-nondeterministic phase:

Definition 9 (Simply (co-)focussed subsystems). *A* FMALL *proof is* focussed *if $\vec{P}$ in D is always singleton. It is* co-focussed *if $\vec{M}$ in $\bar{D}$ is always a singleton. If a proof is both focussed and co-focussed then we say it is* bi-focussed*.*

The notion of 'co-focussing' is not usually possible for (multi-)focussed systems since the invariant of being a singleton is not usually maintained in an asynchronous phase, due to the $\invamp$ rule. However we see $\invamp$ as being deterministic rather than co-nondeterministic, and we can see that the $\&$-rule indeed maintains the invariant of having just one formula on the right of $\Uparrow$.

Theorem 10 (Focussing theorem). *The class of bi-focussed* FMALL*-proofs (*FMALLw*-proofs) is complete for* MALL *(resp.* MALLw*).*

Evidently, this immediately means that FMALL (FMALLw), as well as its focussed and co-focussed subsystems, are also complete for MALL (resp. MALLw). The proof of Thm. 10 follows routinely from any other completeness proof for focussed MALL, e.g. [1,12] ; our only change is at the level of notation.

To aid our exposition, we will sometimes use a 'two-sided' notation and extra connectives so that the intended semantics of sequents are clearer. Strictly speaking, this is just a shorthand for one-sided sequents: the calculi defined in Figs. 1 and 2 are the formal systems we are studying.

Notation 11 *We will write $\Gamma \vdash \Delta$ as shorthand for the sequent $\vdash \overline{\Gamma}, \Delta$, where $\overline{\Gamma}$ is $\{\overline{A} : A \in \Gamma\}$. We extend this notation to annotated sequents in the natural way, writing, e.g., $\Gamma \vdash \Delta \Uparrow \Sigma$ for $\vdash \overline{\Gamma}, \Delta, \Uparrow \Sigma$ and $\Gamma \Downarrow \Delta \vdash \Sigma$ for $\vdash \overline{\Gamma}, \Sigma \Downarrow \overline{\Delta}$. Notice that, in all cases, (co-)foci are always written to the right of a $\Downarrow$ or $\Uparrow$.*

We write $A \multimap B$ as shorthand for the formula $\overline{A} \invamp B$, and $A \multimap^+ B$ as shorthand for the formula $\overline{A} \oplus B$. Sometimes we will write, for example, a step,

$$\multimap_l \frac{\Gamma \vdash \Delta \Downarrow A \quad \Gamma' \Downarrow B \vdash \Delta'}{\Gamma, \Gamma' \Downarrow A \multimap B \vdash \Delta, \Delta'}$$

which, by definition, corresponds to a correct application of $\invamp$ in FMALL*.*

4 An encoding from **CPL2** to **MALLw**

From now on we will work only with MALLw, i.e. affine MALL. In this section we present an encoding of true QBFs into MALLw. The former were also used for the original proof that MALL is **PSPACE**-complete [15,16], though our encoding differs considerably from theirs and leads to a more refined result, cf. Sect. 6.

4.1 Positive and negative encodings of quantifier-free satisfaction

The base cases of our translation from QBFs to MALLw will be quantifier-free Boolean satisfaction. This is naturally a deterministic computation, being polynomial-time computable.[4] However one issue is that this determinism cannot be seen from the point of view of MALLw, since the only deterministic connective ($\invamp$, on the right) is not expressive enough to encode satisfaction.

[4] In fact, formula evaluation is known to be $\mathbf{NC}^1$-complete.

Nonetheless we are able to circumvent this problem since MALLw is at least able to 'see' satisfaction as a problem in **NP** $\cap$ **coNP**, via a pair of encodings corresponding to each class. For non-base levels of **PH** this is morally the same as being deterministic. Indeed, the availability of both types of encodings is the main reason why consider MALLw rather than MALL in this work.

Definition 12 (Positive and negative encodings). *For a quantifier-free Boolean formula φ_0, we define φ_0^- (φ_0^+) as the result of replacing every $\wedge$ in φ_0 for $\&$ (resp. $\otimes$) and every $\vee$ in φ_0 by $⅋$ (resp. $\oplus$).*

For an assignment α and list of variables $\vec{x} = (x_1, \dots, x_k)$, we write $\alpha(\vec{x})$ for the cedent $\{x_i : x_i \in \alpha, i \leq k\} \cup \{\overline{x}_i : x_i \notin \alpha, i \leq k\}$. We write $\alpha^n(\vec{x})$ for the cedent consisting of n copies of each literal in $\alpha(\vec{x})$.

Proposition 13. *Let φ_0 be a quantifier-free Boolean formula with free variables $\vec{x}$ and let α be an assignment. For $n \geq |\varphi_0|$, the following are equivalent:*

1. $\alpha \vDash \varphi_0$.
2. MALLw *proves* $\alpha(\vec{x}) \vdash \varphi_0^-$.
3. MALLw *proves* $\alpha^n(\vec{x}) \vdash \varphi_0^+$.

Proof. $2 \Longrightarrow 1$ and $3 \Longrightarrow 1$ are immediate from the 'soundness' of MALLw with respect to classical logic, by interpreting $\otimes$ or $\&$ as $\wedge$ and $\oplus$ or $⅋$ as $\vee$.

$1 \Longrightarrow 2$ and $1 \Longrightarrow 3$ are both proved by induction on $|\varphi_0|$. In the former case, this follows directly from the invertibility of rules, while in the latter case we appeal to the properties of satisfaction: for $\oplus$-formulas we choose an appropriate disjunct satisfied by α, and for $\otimes$-formulas we split $\alpha^n(\vec{x})$ into $\alpha^k(\vec{x})$ and $\alpha^l(\vec{x})$ s.t. k and l bound the size of their respective conjuncts, reducing to the inductive hypothesis. For both arguments we must appeal to affinity for the base case. $\square$

4.2 Encoding quantifiers in MALLw

As we said before, we do not follow the 'locks-and-keys' approach of [15,16]. Instead we follow a similar approach to Statman's proof that intuitionistic propositional logic is **PSPACE**-hard [23], modulo some improvements that are discussed, for the intuitionistic setting, in [7]. One of the main differences is that we use 'Tseitin extension variables', necessary to avoid an exponential blowup, only in positive positions, not under negation, and this allows our encodings to admit similar proofs to the 'semantic trees' of the QBF we started with.

Definition 14 (Translation from QBFs to MALLw). *Given a QBF $\varphi = Q_k x_k. \cdots .Q_1 x_1.\varphi_0$ with $|\varphi_0| = n$, we define $[\varphi]$ by induction on $k \geq 1$ as follows,*

$$[\varphi_0] \quad := \quad \begin{cases} \varphi^+ & \text{if } Q_1 \text{ is } \exists \\ \varphi^- & \text{if } Q_1 \text{ is } \forall \end{cases}$$

$$[Q_k x_k.\psi] \quad := \quad \begin{cases} ([\psi] \multimap y) \multimap ((x_k^n \multimap y) \oplus (\overline{x}_k^n \multimap y)) & \text{if } Q_k \text{ is } \exists \\ ([\psi] \multimap y) \multimap ((x_k^n \multimap y) \oplus (\overline{x}_k^n \multimap y)) & \text{if } Q_k \text{ is } \forall \end{cases}$$

where y is always fresh.

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{IH}{\alpha^n(\vec{x}), \pm x^n \vdash \vec{y}, [\varphi]}}{\alpha^n(\vec{x}), \pm x^n \vdash \vec{y} \Downarrow [\varphi]}\,{\scriptstyle R_r} \qquad \dfrac{\dfrac{}{y \vdash y}\,{\scriptstyle id}}{\Downarrow y \vdash y}\,{\scriptstyle R_l}}{\alpha^n(\vec{x}), \pm x^n \Downarrow [\varphi] \multimap y \vdash \vec{y}, y}\,{\scriptstyle \multimap_l}}{\alpha^n(\vec{x}), \pm x^n, [\varphi] \multimap y \vdash \vec{y}, y}\,{\scriptstyle D_l}}{\alpha^n(\vec{x}), [\varphi] \multimap y \vdash \vec{y}, \pm x^n \multimap y}\,{\scriptstyle \multimap_r}}{\alpha^n(\vec{x}), [\varphi] \multimap y \vdash \vec{y} \Downarrow \pm x^n \multimap y}\,{\scriptstyle R_r}}{\alpha^n(\vec{x}), [\varphi] \multimap y \vdash \vec{y} \Downarrow (x^n \multimap y) \oplus (\overline{x}^n \multimap y)}\,{\scriptstyle \oplus_r}}{\alpha^n(\vec{x}), [\varphi] \multimap y \vdash \vec{y}, (x^n \multimap y) \oplus (\overline{x}^n \multimap y)}\,{\scriptstyle D_r}}{\alpha^n(\vec{x}) \vdash \vec{y}, ([\varphi] \multimap y) \multimap ((x^n \multimap y) \oplus (\overline{x}^n \multimap y))}\,{\scriptstyle \multimap_r}}{\alpha^n(\vec{x}) \vdash \vec{y}, [\exists x.\varphi]}\,{\scriptstyle =}
$$

Fig. 3. Proof of $\exists$ case for left-right direction of Lemma 15.

Lemma 15. *Let $\varphi(\vec{x})$ be a QBF with all free variables displayed and with matrix φ_0. Then $\alpha \vDash \varphi$ if and only if* MALLw *proves $\alpha^n(\vec{x}) \vdash \vec{y}, [\varphi]$ for any $n \geq |\varphi_0|$, any assignment α and any $\vec{y}$ disjoint from $\vec{x}$.*

Proof (sketch). We proceed by induction on the quantifier complexity of φ. For the base case, when φ is quantifier-free, we appeal to Prop. 13; the left-right direction follows directly by weakening (cf. Prop. 7), while the right-left direction follows after observing that $\vec{y}$ does not occur in $[\varphi]$ or $\alpha^n(\vec{x})$, and so may be systematically deleted from any proof while preserving correctness.

For the inductive step, in the left-right direction we give appropriate bi-focussed proofs in Figs. 3 and 4, where $\pm x$ in Fig. 3 is chosen to be x if $x \in \alpha$ and $\overline{x}$ otherwise, the derivations marked *IH* are obtained by the inductive hypothesis, and the derivation marked ... in Fig. 4 is analogous to the one on the left of it. For the right-left direction, we need only consider the other possibilities that could occur during bi-focussed proof search, by the focussing theorem, Thm. 10. For the $\exists$ case, bottom-up, one could have chosen to first decide on $[\varphi] \multimap y$ in the antecedent. The consequent $\multimap_l$ step would have to send the formula $(x^n \multimap y) \oplus (\overline{x}^n \multimap y)$ to the premiss for y, since otherwise every variable occurrence in that premiss would be distinct and there would be no way to correctly finish proof search. Thus, possibly after weakening, we may apply the inductive hypothesis to the other premiss. A similar analysis of the upper $\multimap_l$ step in Fig. 3 means that any other split will allow us to appeal to the inductive hypothesis after weakening. For the $\forall$ case the argument is much simpler, since no matter which order we 'co-decide', we will end up with the same leaves.[5] (Note that, for the derivations for the innermost quantifier ($\exists$ or $\forall$), the topmost R or $\bar{R}$ step of Figs. 3 or 4 (resp.) does not occur.) $\square$

[5] This is actually exemplary of the more general phenomenon that invertible phases of rules are 'confluent'.

$$
=\,\dfrac{\multimap_r\,\dfrac{\bar{D}_r\,\dfrac{\&_r\,\dfrac{\bar{R}_r\,\dfrac{\multimap_r\,\dfrac{\bar{D}_l\,\dfrac{\multimap_l^+\,\dfrac{\bar{R}_r\,\dfrac{\overset{\mathit{IH}}{\alpha^n(\vec{x}), x^n \vdash \vec{y}, y, [\varphi]}}{\alpha^n(\vec{x}), x^n \vdash \vec{y}, y \Uparrow [\varphi]} \qquad \bar{R}_l\,\dfrac{id\,\dfrac{}{\alpha^n(\vec{x}), x^n, y \vdash \vec{y}, y}}{\alpha^n(\vec{x}), x^n \Uparrow y \vdash \vec{y}, y}}{\alpha^n(\vec{x}), x^n \Uparrow [\varphi] \multimap^+ y \vdash \vec{y}, y}}{\alpha^n(\vec{x}), x^n, [\varphi] \multimap^+ y \vdash \vec{y}, y}}{\alpha^n(\vec{x}), [\varphi] \multimap^+ y \vdash \vec{y}, x^n \multimap y}}{\alpha^n(\vec{x}), [\varphi] \multimap^+ y \vdash \vec{y} \Uparrow x^n \multimap y} \qquad \bar{R}_r\,\dfrac{\vdots}{\alpha^n(\vec{x}), [\varphi] \multimap^+ y \vdash \vec{y} \Uparrow \overline{x}^n \multimap y}}{\alpha^n(\vec{x}), [\varphi] \multimap^+ y \vdash \vec{y} \Uparrow (x^n \multimap y) \& (\overline{x}^n \multimap y)}}{\alpha^n(\vec{x}), [\varphi] \multimap^+ y \vdash \vec{y}, (x^n \multimap y) \& (\overline{x}^n \multimap y)}}{\alpha^n(\vec{x}) \vdash \vec{y}, ([\varphi] \multimap^+ y) \multimap ((x^n \multimap y) \& (\overline{x}^n \multimap y))}}{\alpha^n(\vec{x}) \vdash \vec{y}, [\forall x.\varphi]}
$$

Fig. 4. Proof of $\forall$ case for left-right direction of Lemma 15.

Theorem 16. *Let φ be a closed QBF. $\vDash \varphi$ if and only if* MALLw *proves* $[\varphi]$.

Proof. Follows immediately from Lemma 15. □

5 Focussed proof search as alternating time predicates

In this section we will show how to express focussed proof search as an alternating polynomial-time predicate that will later allow us to calibrate the complexity of proof search with levels of the QBF and polynomial hierarchies.

The following definition generalises the notions of 'decide depth' and 'release depth' found in other works, e.g. [20].

Definition 17 (Nondeterministic and co-nondeterministic complexity). *For a proof $\mathcal{P}$, its* nondeterministic *complexity, $\sigma(\mathcal{P})$, is the least number of alternations between D and $\bar{D}$ steps in a branch through $\mathcal{P}$, starting with D. Its* co-nondeterministic *complexity, $\pi(\mathcal{P})$, is defined similarly, only starting with $\bar{D}$.*

For a cedent Γ, we write $\sigma(\Gamma)$ ($\pi(\Gamma)$) for the least $k \in \mathbb{N}$ such that there is a proof $\mathcal{P}$ of $\vdash \Gamma$ with $\sigma(\mathcal{P}) \leq k$ (resp. $\pi(\mathcal{P}) \leq k$).

Notice that the above notions of complexity are robust under the choice of multi-focussed, (co-)focussed or bi-focussed proof systems: while the number of D or $\bar{D}$ steps may increase, the number of alternations remains constant. This robustness will also apply to the other concepts we introduce in this section.

We will now introduce 'provability predicates' that delineate the complexity of proof search in a similar way to the QBF and polynomial hierarchies we presented earlier. Recall the notions of deterministic, nondeterministic and co-nondeterministic rules from Dfn. 8, cf. Fig. 2.

Definition 18 (Focussing hierarchy). *A cedent Γ is:*

- Σ^f_1-provable *if there is a proof of $\vdash \Gamma$ using only deterministic and nondeterministic steps.*
- Π^f_1-provable *if every maximal path from $\vdash \Gamma$, bottom-up, through deterministic and co-nondeterministic rules ends at a correct initial sequent.*
- Σ^f_{k+1}-provable *if there is a derivation of $\vdash \Gamma$, using only deterministic and nondeterministic steps, from sequents $\vdash \Gamma_i$ which are Π^f_k-provable.*
- Π^f_{k+1}-provable *if every maximal path from $\vdash \Gamma$, bottom-up, through deterministic and co-nondeterministic rules ends at a Σ^f_k-provable sequent.*

As expected, we may directly link the (co-)nondeterministic complexity of a cedent with its position in the 'focussing hierarchy':

Proposition 19. *A cedent Γ is Σ^f_k-provable (Π^f_k-provable) if and only if $\sigma(\Gamma) \leq k$ (resp. $\pi(\Gamma) \leq k$).*

Moreover, we have a natural correspondence between the focussing hierarchy and the other hierarchies we have discussed:

Theorem 20. *Σ^f_k-provability (Π^f_k-provability) is computable in Σ^p_k (resp. Π^p_k). Moreover, for $k > 0$, there are Σ^q_k (resp. Π^q_k) formulas $\Sigma^f_k\text{-}\mathsf{Prov}_n$ (resp. $\Pi^f_k\text{-}\mathsf{Prov}_n$), constructible in time polynomial in $n \in \mathbb{N}$, that compute the Σ^f_k-provability (resp. Π^f_k-provability) on all formulas A such that $|A| = n$.*

We omit a proof of this, which is routine, due to space constraints, but direct the reader to the analogous construction in previous work, [7]. We point out that, for the $\otimes$ rule, even though there are two premisses, the rule is context-splitting, and so a nondeterministic machine may simply split into two parallel threads with no blowup in complexity.

6 An 'inverse' encoding from MALLw into CPL2

From Thm. 20, let us henceforth fix appropriate QBFs $\Sigma^f_k\text{-}\mathsf{Prov}_n$ and $\Pi^f_k\text{-}\mathsf{Prov}_n$ computing Σ^f_k-provability and Π^f_k-provability, resp., for $k > 0$. Given these formulas, we will in this section give an explicit encoding from MALLw to CPL2, i.e. a polynomial-time function from MALLw-formulas to QBFs whose restriction to theorems has image in CPL2. Moreover, we will show that this encoding acts as an 'inverse' to the one we gave in Sect. 4, and finally identify fragments of MALLw complete for each level of **PH**.

To this end, the issue with the complexity functions σ, π introduced earlier is that they are hard to compute. Instead we give an 'over-estimate' here that will suffice for the encodings we are after.

So that the notions we define below are well defined, we will assume some arbitrary total order on formulae. The precise choice is unimportant, as long as it is polynomial-time computable; this way our ultimate encoding remains computable in polynomial time.

$$
\begin{array}{rcll}
\lceil\sigma\rceil(\vec{a}) & := & 0 & \\
\lceil\sigma\rceil(\Gamma, A \parr B) & := & \lceil\sigma\rceil(\Gamma, A, B) & A \text{ is least in } \Gamma, A \\
\lceil\sigma\rceil(\vec{a}, \vec{P}, P) & := & \lceil\sigma\rceil(\vec{a}, \vec{P} \Downarrow P) & P \text{ is least in } \vec{P}, P \\
\lceil\sigma\rceil(\vec{a}, \vec{P}, \vec{M}, M) & := & 1 + \lceil\pi\rceil(\vec{a}, \vec{P}, \vec{M}, \Uparrow M) & M \text{ is least in } \vec{M}, M \\
\\
\lceil\pi\rceil(\vec{a}) & := & 0 & \\
\lceil\pi\rceil(\Gamma, A \parr B) & := & \lceil\pi\rceil(\Gamma, A, B) & A \text{ is least in } \Gamma, A \\
\lceil\pi\rceil(\vec{a}, \vec{P}, P) & := & 1 + \lceil\sigma\rceil(\vec{a}, \vec{P} \Downarrow P) & P \text{ is least in } \vec{P}, P \\
\lceil\pi\rceil(\vec{a}, \vec{P}, \vec{M}, M) & := & \lceil\pi\rceil(\vec{a}, \vec{P}, \vec{M}, \Uparrow M) & M \text{ is least in } \vec{M}, M
\end{array}
$$

$$
\begin{array}{rcll}
\lceil\sigma\rceil(\Gamma \Downarrow A \oplus B) & := & \begin{cases} \lceil\sigma\rceil(\Gamma, A) & \lceil\sigma\rceil(A) \geq \lceil\sigma\rceil(B) \\ \lceil\sigma\rceil(\Gamma, B) & \text{otherwise} \end{cases} & \\
\lceil\sigma\rceil(\Gamma \Downarrow A \otimes B) & := & \begin{cases} \lceil\sigma\rceil(\Gamma, A) & \lceil\sigma\rceil(A) \geq \lceil\sigma\rceil(B) \\ \lceil\sigma\rceil(\Gamma, B) & \text{otherwise} \end{cases} & \\
\lceil\sigma\rceil(\Gamma \Downarrow X) & := & \lceil\sigma\rceil(\Gamma, X) & X \text{ is } a \text{ or } N
\end{array}
$$

$$
\begin{array}{rcll}
\lceil\pi\rceil(\Gamma \Uparrow A \,\&\, B) & := & \begin{cases} \lceil\pi\rceil(\Gamma, A) & \lceil\pi\rceil(A) \geq \lceil\pi\rceil(B) \\ \lceil\pi\rceil(\Gamma, B) & \text{otherwise} \end{cases} & \\
\lceil\pi\rceil(\Gamma \Uparrow X) & := & \lceil\pi\rceil(\Gamma, X) & X \text{ is } O \text{ or } P
\end{array}
$$

Fig. 5. Approximating (co-)nondeterminstic complexities.

Definition 21 (Approximating the complexity of a sequent). *We define the functions $\lceil\sigma\rceil$ and $\lceil\pi\rceil$ on sequents in Fig. 5.*

It is not hard to see that we have:

Proposition 22 (Over-estimatation). *$\sigma \leq \lceil\sigma\rceil$ and $\pi \leq \lceil\pi\rceil$.*

Notice that the over-estimation for the $\otimes$ case is particularly extreme: in the worst case we have that the entire context is copied to one branch. This, along with the fact that the base case applies to only atomic cedents, means that it does not actually matter which order we compute an approximation.

Moreover, we have the following:

Proposition 23. *$\lceil\sigma\rceil$ and $\lceil\pi\rceil$ are polynomial-time computable.*

Finally, we have that these approximations are in fact tight for the translation $[\cdot]$ from MALLw-formulas to QBFs (cf. Sect. 4) by an inspection of its definition:

Proposition 24. *$\lceil\sigma\rceil([\varphi]) = \sigma([\varphi])$ and $\lceil\pi\rceil([\varphi]) = \pi([\varphi])$.*

We are now ready to define our 'inverse' encoding to $[\cdot]$:

Definition 25 (Encoding). *For a MALLw formula A, we define $\langle A\rangle$ as follows:*

$$
\langle A\rangle \;:=\; \begin{cases} \Sigma^f_k\text{-}\mathsf{Prov}_{|A|}(A) & \textit{if } k = \lceil\sigma\rceil(A) \leq \lceil\pi\rceil(A) \\ \Pi^f_k\text{-}\mathsf{Prov}_{|A|}(A) & \textit{if } k = \lceil\pi\rceil(A) < \lceil\sigma\rceil(A) \end{cases}
$$

Finally, we are able to present our main result:

Theorem 26. *We have the following:*

1. $[\cdot]$ *is an encoding from* $\mathsf{CPL2}$ *to* MALLw.
2. $\langle\cdot\rangle$ *is an encoding from* MALLw *to* $\mathsf{CPL2}$.
3. *The composition* $\langle\cdot\rangle \circ [\cdot] : \mathsf{CPL2} \to \mathsf{CPL2}$ *preserves quantifier complexity, i.e. it maps* Σ^q_k *(*Π^q_k*) theorems to* Σ^q_k *(resp.* Π^q_k*) theorems, for* $k > 0$.

Proof. We have already proved 1 in Thm. 16. 2 follows from the definitions of Σ^f_k-Prov and Π^f_k-Prov (cf. Thm. 20), under Props. 19 and 22. Finally 3 then follows by tightness of the approximation in the image of $[\cdot]$, Prop. 24.

Consequently, we may identify polynomial-time recognisable subsets of MALLw-formulas whose theorems are complete for levels of the polynomial hierarchy:

Corollary 27. *We have the following, for* $k > 0$*:*

1. $\{A \in \mathsf{MALLw} : \lceil\sigma\rceil(A) \leq k\}$ *is* Σ^p_k*-complete.*
2. $\{A \in \mathsf{MALLw} : \lceil\pi\rceil(A) \leq k\}$ *is* Π^p_k*-complete.*

7 Conclusions and further remarks

We gave a refined presentation of (multi-)focussed systems for multiplicative-additive linear logic, and its affine variant, that accounts for deterministic computations in proof search, cf. Sect. 3. We showed that it admits rather controlled normal forms in the form of *bi-focussed* proofs, and highlighted a duality between focussing and 'co-focussing' that emerges thanks to this presentation.

The main reason for using focussed systems such as ours was to better reflect the alternating complexity of bottom-up proof search, cf. Sect. 5. We justified the accuracy of these bounds by showing that natural measures of proof search complexity for FMALLw tightly delineate the theorems of MALLw according to associated levels of the polynomial hierarchy, cf. Sects. 4 and 6. These results exemplify how the capacity of proof search to provide optimal decision procedures for logics may extend to important subclasses of **PSPACE**.

It is natural to wonder whether a similar result to Thm. 26 could be obtained for (non-affine) MALL. The reason we chose MALLw is that it allows for a robust and uniform approach that highlights the ability of focussed systems to realise tight alternating time bounds for logics, without too many extraneous technicalities. Nonetheless, we briefly discuss how a similar result could be obtained for MALL, although stop short of giving formal results due to space constraints.

The main issue for MALL is the fact that there does not seem to be any 'negative' encoding of quantifier-free satisfaction, apparently only allowing characterisations of the levels Σ^p_{2k+1} and Π^p_{2k} in the same way. Apart from this, the rest of the argument can be recovered for MALL with some local adaptations. One may redefine $[\forall x.\varphi]$ as $(x \oplus \overline{x}) \multimap [\varphi]$, in order to avoid the need for weakening, and the associated coding of assignments also needs to be more structured, combining $\otimes$ and $\&$ to reflect the precise choices made in proof search. The proof of the corresponding form of Lemma 15 requires a more global analysis, for the right-left direction, due to the absence of weakening. For the inverse encoding, the definition of $\langle\cdot\rangle$ remains the same, and our main inversion result, Thm. 26, goes through as before.

In fact, by enriching the proof system with a deterministic 'evaluation' rule for positive encodings of quantifier-free satisfaction, we may recover fragments of MALL complete for *each* level of **PH**. A similar approach was followed for fragments of intuitionistic logic in [7], although this leads to further technicalities when approximating (co-)nondeterministic complexity of a sequent.

Our presentation of FMALL should extend to logics with units/constants, quantifiers and exponentials, following traditional approaches to focussed linear logic, cf. [1,12]. It would be interesting to see what could be said about the complexity of proof search for such logics. For instance, the usual $\forall$ rule becomes deterministic in our analysis, since it does not branch:

$$\forall\,\frac{\Gamma, A(y)}{\Gamma, \forall x. A(x)}\; y \text{ is fresh}$$

As a result, the alternating complexity of proof search is not affected by the $\forall$-rule, but rather interactions between positive connectives, including $\exists$, and negative connectives such as $\&$. Interpreting this over a classical setting could give us new ways to delineate true QBFs according to the polynomial hierarchy, determined by the alternation of $\exists$ and propositional connectives rather than $\forall$.

Much of the literature on *logical frameworks* via focussed systems is based around the idea that an inference rule may be simulated by a 'bipole', i.e. a single alternation between an invertible and non-invertible phase of inference steps. However, accounting for determinism might yield more refined simulations, where non-invertible rules are simulated by phases of deterministic and nondeterministic rules, but not co-nondeterministic ones, cf. Dfn. 18. In particular we envisage this to be possible for certain translations between modal logic and first-order logic, cf. [18,17].

Acknowledgements. I would like to thank Taus Brock-Nannestad, Kaustuv Chaudhuri, Sonia Marin and Dale Miller for many fruitful discussions about focussing, and in particular, on the presentation of it herein.

References

1. J. Andreoli. Logic programming with focusing proofs in linear logic. *J. Log. Comput.*, 2(3):297–347, 1992.
2. S. R. Buss and R. Iemhoff. The depth of intuitionistic cut free proofs, 2003.
3. A. K. Chandra, D. C. Kozen, and L. J. Stockmeyer. Alternation. *J. ACM*, 28(1):114–133, Jan. 1981.

4. K. Chaudhuri, D. Miller, and A. Saurin. Canonical sequent proofs via multi-focusing. In *Fifth IFIP International Conference On Theoretical Computer Science - TCS 2008, IFIP 20th World Computer Congress, TC 1, Foundations of Computer Science, September 7-10, 2008, Milano, Italy*, pages 383–396, 2008.
5. S. Cook and P. Nguyen. *Logical Foundations of Proof Complexity.* Cambridge University Press, New York, NY, USA, 1st edition, 2010.
6. S. A. Cook and R. A. Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44(1):36–50, 1979.
7. A. Das. Alternating time bounds from variants of focussed proof systems, 2017. Preprint. `http://anupamdas.com/alt-time-bnds-var-foc-sys.pdf`.
8. O. Delande, D. Miller, and A. Saurin. Proof and refutation in MALL as a game. *Ann. Pure Appl. Logic*, 161(5):654–672, 2010.
9. J. Girard. Linear logic. *Theor. Comput. Sci.*, 50:1–102, 1987.
10. A. Heuerding, M. Seyfried, and H. Zimmermann. Efficient loop-check for backward proof search in some non-classical propositional logics. In *International Workshop on Theorem Proving with Analytic Tableaux and Related Methods*, pages 210–225. Springer, 1996.
11. J. Krajíček. *Bounded arithmetic, propositional logic, and complexity theory.* Cambridge University Press, New York, NY, USA, 1995.
12. O. Laurent. *A study of polarization in logic.* Theses, Université de la Méditerranée - Aix-Marseille II, Mar. 2002.
13. R. Letz. Lemma and model caching in decision procedures for quantified boolean formulas. In *Automated Reasoning with Analytic Tableaux and Related Methods, International Conference, TABLEAUX 2002, Copenhagen, Denmark, July 30 - August 1, 2002, Proceedings*, pages 160–175, 2002.
14. C. Liang and D. Miller. Focusing and polarization in linear, intuitionistic, and classical logics. *Theoretical Computer Science*, 410(46):4747–4768, 2009.
15. P. Lincoln, J. C. Mitchell, A. Scedrov, and N. Shankar. Decision problems for propositional linear logic. In *31st Annual Symposium on Foundations of Computer Science, Volume II*, pages 662–671, 1990.
16. P. Lincoln, J. C. Mitchell, A. Scedrov, and N. Shankar. Decision problems for propositional linear logic. *Ann. Pure Appl. Logic*, 56(1-3):239–311, 1992.
17. S. Marin, D. Miller, and M. Volpe. A focused framework for emulating modal proof systems. In *11th Conference on "Advances in Modal Logic,"*, pages 469–488, 2016.
18. D. Miller and M. Volpe. Focused labeled proof systems for modal logic. In *Logic for Programming, Artificial Intelligence, and Reasoning - 20th International Conference, LPAR-20*, pages 266–280, 2015.
19. S. Negri, J. Von Plato, and A. Ranta. *Structural proof theory.* Cambridge University Press, 2008.
20. V. Nigam. Investigating the use of lemmas. 2007. Preprint.
21. H. Ono. *Proof-Theoretic Methods in Nonclassical Logic –an Introduction*, volume 2 of *MSJ Memoirs*, pages 207–254. The Mathematical Society of Japan, Tokyo, Japan, 1998.
22. C. H. Papadimitriou. *Computational complexity.* Addison-Wesley, 1994.
23. R. Statman. Intuitionistic propositional logic is polynomial-space complete. *Theor. Comput. Sci.*, 9:67–72, 1979.
24. L. J. Stockmeyer. The polynomial-time hierarchy. *Theor. Comput. Sci.*, 3(1):1–22, 1976.
25. A. Troelstra and H. Schwichtenberg. *Basic Proof Theory*, volume 43 of *Cambridge Tracts in Theoretical Computer Science.* Cambridge University Press, 1996.