The complexity of propositional proofs in deep inference

submitted by

Anupam Das

for the degree of PhD

of the

University of Bath

Department of Computer Science

October 2014

Summary

Deep inference is a relatively recent proof methodology whose systems differ from traditional systems by allowing inference rules to operate on any connective appearing in a formula, rather than just the main connective. Its distinguishing feature, from a structural proof theoretic point of view, is that its systems are *local*: inference steps can be checked in constant time, a property impossible to achieve in Gentzen sequent calculi.

Due to the greater flexibility in applying inference rules, deep inference systems introduce more normal forms to classical proof theory, splitting Gentzen cut-elimination into smaller steps. While the complexity of full cut-elimination in the sequent calculus is necessarily exponential in the size of the input proof, these intermediate procedures exhibit varying complexities, while still entailing properties of proof theoretic interest, such as consistency and decidability.

This dissertation contributes to the classification of these intermediate classes of proof, from the point of view of proof complexity, and introduces new techniques to analyse their complexity.

Acknowledgements

I am indebted to my advisor, Alessio Guglielmi, for all the time and patience he has afforded me over my tenure as a PhD student. The last three years have been a unique and wonderful experience. To any prospective students reading this dissertation, I would encourage you to contact Alessio and hear what he has to say.

I am also grateful to the Logic and Semantics group at Bath - Guy, Jim, Cai, Pierre, Paola, Martin, Ana, Alvin, Willem, Etienne, John and Alessio - with whom a great deal of mathematics was discussed over lunch, seminars and beers, not usually related to research but nonetheless very worthwhile.

Of course, I am grateful to my family for everything, in particular my sister Payel for being a wonderful sibling and an excellent role model. I would also like to thank Catrin for so many entertaining conversations about logic (and everything else) and with whom I grew as a mathematician.

A special thanks to Paul and Pip, the Bird-Kings, two of my dearest friends, for letting me stay with them in Bristol while I was writing this dissertation. Congratulations on getting married and I'm sorry that my motorbike is still at yours.

Finally, I would like to thank all my coaches and training partners over these last few years for giving me something other than maths to be passionate and geeky about. I had some great times (in every sense) with Dave, Matt, Jake, the Chef and the Beast, who were all excellent training partners, rivals and friends. Thank you so much to coach Martin and the rest of TBAC for supporting me, and also to Chris and Keith from Bristol & West for being so welcoming when I was in Bristol.

Contents

Chapter 1

Introduction

Proof theory is pursued today from many different viewpoints. Structural proof theory, broadly speaking, is the study of normal forms of proofs, often with the purpose of exposing properties of computational interest for the associated logic. The most prominent example is perhaps the notion of *cut-free* proof in Gentzen's sequent calculus. Such a proof contains no instances of the cut rule,

$$\frac{\Gamma \to \Delta, A \quad A, \Sigma \to \Pi}{\Gamma, \Sigma \to \Delta, \Pi}$$

and so contains only subformulae of its conclusion. From Gentzen's celebrated Hauptsatz [Gen35] we know that any sequent proof can be transformed into cut-free form (albeit at an exponential cost in complexity) and from here it is not difficult to infer consistency, decidability and Craig interpolation for, say, classical propositional logic. These particular normal forms are exemplary of *analytic* proofs, ones that involve only concepts found in their conclusion.[1]

Proof complexity focusses on the relative strength of formal systems, measured in terms of the complexity of their proofs. The basic relation between proof systems is that of *polynomial simulation*: a system P polynomially simulates a system Q if every Q-proof can be polynomially transformed into a P-proof of the same conclusion. This is not dissimilar to how theories are compared by theorem provability in proof theory, and indeed is fundamentally related to such questions for theories of *bounded arithmetic* via certain propositional translations [PW81] [Coo75]. While of independent proof theoretic interest, proof complexity is also fundamentally linked with important

[1]What exactly constitutes an analytic proof is the subject of debate. A particularly relevant issue is what is meant here by 'concept', on which deep inference takes a more liberal stance than the sequent calculus [BG09a].

questions of complexity theory, P vs. NP and coNP vs. NP, via the celebrated Cook-Levin theorem that the set of propositional tautologies is coNP-complete [Coo71].

The complexity of weak systems typically studied by structural proof theorists, such as cut-free sequent calculi, is well-understood; in particular there are known exponential lower bounds on the size of proofs for these systems. Proof complexity typically concerns itself with systems not much weaker than Hilbert-Frege systems, or equivalently Gentzen sequent calculi with cut. Consequently, proof complexity and structural proof theory, for the most part, focus on different classes of proof systems.

In this dissertation we focus on *deep inference* systems, which we argue lies at an intersection of the interests of both areas. While originally motivated by structural proof theoretic considerations, work in recent years has highlighted nontrivial complexity properties of these systems, and this forms the subject matter of the present work.

From the point of view of structural proof theory the distinguishing feature of deep inference systems is that they are *local*, i.e. inference steps can be checked in constant time [BT01]. *Structural* rules, ones that introduce, destroy or duplicate formulae, operate only on atoms, and the remaining rules (called *logical* rules) are *linear*, i.e. each formula variable occurs exactly once in the premiss and conclusion. The ability to reduce structural rule steps to atomic form in deep inference crucially relies on the following features:

1. Inference rules can operate on any connective occurring in a formula, not just the main connective.

2. Logical rules are suitably chosen to allow for this reduction. In particular the *medial* rule,
$$\frac{(A \wedge B) \vee (C \wedge D)}{(A \vee C) \wedge (B \vee D)}$$
plays an essential role in localising contraction.

Notice that the medial rule does not obey the subformula property, and so has no analogue in the sequent calculus. In fact this sort of locality is impossible to achieve in the sequent calculus since the contraction rule cannot be reduced to atomic form [Brü03].

Due to the greater flexibility in designing derivations and atomicity of the structural rules we gain a rich theory of proofs in deep inference. Structural inferences generally commute with logical inferences and overlaps between structural steps can be manipulated in a uniform way. Normalisation procedures in deep inference break down traditional cut-elimination into smaller subprocedures, equivalent to eliminating only cuts between descendants of certain structural inferences [Brü06] [Jeř09], yielding finer notions of analyticity for classical proofs. The induced normal forms share many of the structural properties of cut-free sequent proofs, e.g. from their existence it is simple to infer consistency and decidability of the system.

At the same time these intermediate classes of proof exhibit complexity properties in between those of Gentzen sequent calculi with and without cut. While it is of independent interest to examine exactly which interactions cause the exponential blowup in cut-elimination, the approach of restricting cut-steps is also prevalent in proof complexity where researchers have, for example, studied restricting cut-formulae to ones of bounded depth, e.g. in [Kra95], or ones free of negation e.g. in [AGG00]. The overall aim of this research direction is to gain insight into how one might eventually find lower bounds for Hilbert-Frege systems, the existence of which has remained an open problem in proof complexity for decades. In this way we propose that studying the complexity of normal forms of deep inference proofs contributes towards this programme of research.

As a contribution in another direction, it is argued that the deep inference methodology allows for proofs that feature less redundancy and arbitrary syntactic dependencies than traditional formalisms, due to the greater flexibility in proof design [GGP10] [BL05]. Eliminating this so-called *syntactic bureaucracy* [Gir87] is an important step towards understanding the *identity of proofs*, an issue arising from what is known as Hilbert's 24th problem [Thi03]. We argue that this hypothesis, that deep inference proofs feature less syntactic bureaucracy, is supported by results in this and previous work, where nontrivial improvements in complexity are exhibited for analytic deep inference systems against their traditional counterparts.

Overview of dissertation

In Part I we introduce the basic systems of deep inference for classical propositional logic, namely the system SKS and its subsystems KS^+ and KS, the so-called *analytic* fragments of SKS [BG09b]. The crucial difference between KS and KS^+, from the point of view of complexity, is the presence in KS^+ of the *cocontraction* rule, $\dfrac{A}{A \wedge A}$. This rule can be considered to simulate dag-like behaviour in proofs since it allows us to "reuse" the already proved formula A. Consequently a naïve elimination procedure for cocontraction can result in an exponential blowup in the size of a proof.

While it is simple to see that SKS is equivalent to Hilbert-Frege systems, from the point of view of complexity, it was an early observation in deep inference that KS^+

is equivalent to the class of Gentzen sequent proofs where there are no cuts between descendants of left negation and right negation steps, while KS is equivalent to the class that further has no cuts between descendants of any structural inferences [Brü06].

It turns out that restricting our attention to these classes of cuts is an interesting pursuit from the point of view of proof complexity. In Part II we consider the complexity of analytic systems with cocontraction, namely KS^+ and its variants. Due to the aforementioned correspondence it follows that KS^+ is polynomially equivalent to the tree-like fragment of the monotone sequent calculus. By a result of Atserias et al. in [AGP02] it follows that KS^+ quasipolynomially[2] simulates Hilbert-Frege systems, first observed by Jeřábek in [Jeř09]. This construction was internalised to deep inference by Bruscoli et al. in [BGGP10] although we give a proof here for completeness, which is just a simplified version of those that have already appeared in the literature.

We also address the effect of limiting the *depth* at which inference rules may operate. A simple observation yields that one can restrict every inference step to operate at the 'surface' of a formula, if non-atomic structural inferences are permitted, with only polynomial blowup in complexity. We notice that the resulting system is polynomially equivalent to augmenting a cut-free sequent calculus[3] with elimination rules for the connectives:

$$\frac{\Gamma, A \vee B}{\Gamma, A, B} \qquad \frac{\Gamma, A \wedge B}{\Gamma, A} \qquad \frac{\Gamma, A \wedge B}{\Gamma, B} \qquad \frac{\Gamma, \bot}{\Gamma}$$

By the aforementioned results we obtain a quasipolynomial-time reduction from Hilbert-Frege provability to provability in the cut-free sequent calculus augmented with these elimination rules. We propose that finding lower bounds for this system could be easier than doing so directly for Hilbert-Frege systems, due to its relative simplicity and the controlled way in which it breaks the subformula property.

The rest of this work studies the complexity of KS and related systems, by examining the complexity of eliminating cocontraction steps in KS^+-proofs.

In Part III we analyse separately the structural and logical fragments of deep inference systems in order to gain a better understanding of their effect on complexity. We should point out that such an analysis is possible due to the locality of deep inference, since structural rule steps generally commute with logical steps, allowing for the two fragments to be studied (and indeed varied) independently. In the sequent calculus this is not possible due to nontrivial interactions between the logical and structural rules.

We introduce *atomic flows*, graphs that trace structural changes in proofs, i.e.

[2]A quasipolynomial in n is a function of size $n^{\log^{\Theta(1)} n}$, although this particular simulation has complexity $n^{\Theta(\log n)}$.

[3]It is crucial here that proofs are permitted to be in dag form.

creation, destruction and duplication of atoms, and define certain rewriting systems on these graphs. We show that these systems induce normalisation procedures from KS^+-proofs to KS-proofs and examine the complexity of optimal reduction strategies. In fact, since we only consider structural interactions, these procedures are independent of the choice of logical rules in the system. Atomic flows and similar procedures first appeared in [GG08], although we only give a brief account of atomic flows, focussing instead on the complexity of these procedures. A formal account of normalisation via flows can be found in [Gun09].

The logical fragment is analysed as a term rewriting system, given by the two rules below,[4] and we show that this fragment does not contribute superpolynomially to the overall complexity of a proof, even in the presence of equations for the units $\top, \bot$.

$$\begin{aligned} A \bullet (B \circ C) &\rightarrow (A \bullet B) \circ C \\ (A \bullet B) \circ (C \bullet D) &\rightarrow (A \circ C) \bullet (B \circ D) \end{aligned}$$

However this fragment is rich enough to encode all the information in a KS-proof, from the point of view of complexity, in the sense that all KS-proofs can be put into a normal form where the choice and configuration of structural rule steps is determined. In other words, one can consider proof size to be determined by the structural fragment of deep inference, whereas the complexity of proof search is determined by the logical fragment. This is explained in more detail in Chapt. 7

Using a well-known trick in deep inference we obtain another reduction from Hilbert-Frege provability: we give a polynomial-time reduction from the question "is there a Hilbert-Frege proof of a tautology τ of size S?" to the question "is there a rewrite path from A to B?", where A and B are formulae dependent on τ and S. Interestingly, Straßburger has found polynomial-time combinatorial characterisations of these two rules separately [Str07], and so there is already some understanding of the complexity of rewriting in this system. Thus we argue that studying this rewriting system might prove a fruitful approach towards finding lower bounds for Hilbert-Frege systems.

In Part IV we use previous results, namely the normalisation procedures induced by flow rewriting, to gain some insight into the complexity of KS. We give polynomial simulations of truth tables, and also exponential separations from cut-free sequent systems, Resolution systems, and bounded-depth Hilbert-Frege systems. Finally we construct quasipolynomial-size proofs of the propositional pigeonhole principle and related classes of tautologies, demonstrating the relative strength of KS.

[4]The term rewriting system induced by these two rules operate modulo associativity and commutativity of the two binary operators $\bullet$ and $\circ$. Intuitively, $\bullet$ represents $\wedge$ and $\circ$ represents $\vee$.

Part I

Preliminaries

Chapter 2

Propositional proof complexity

In this chapter we present basic concepts in proof complexity. Some of our definitions differ slightly from those occurring in the literature, so that they are compatible with notions that appear later when we specialise to deep inference systems, but fundamentally everything in this chapter is standard. A more thorough introduction to the subject can be found in, for example, [Kra95].

2.1 The language of propositional logic

The language of propositional logic consists of a countable set of atoms, or propositional variables, a, b etc. and their duals $\bar{a}, \bar{b}$ etc. and the connectives $\wedge, \vee, \top, \bot$, all with their usual interpretations. Formulae, denoted A, B etc., are built freely over this language in the usual way, and we denote by the symbol $\equiv$ syntactic equivalence of expressions over this language.

Notice that we do not have any symbol for negation in our language. Instead formulae are in negation normal form. We may write $\bar{A}$ to denote the De Morgan dual of a formula A, obtained by the following rules:

$$\bar{\top} \equiv \bot \quad , \quad \bar{\bot} \equiv \top \quad , \quad \bar{\bar{a}} \equiv a \quad , \quad \overline{A \wedge B} \equiv \bar{A} \vee \bar{B} \quad , \quad \overline{A \vee B} \equiv \bar{A} \wedge \bar{B}$$

The connectives $\bot$ and $\top$ are called constants or units. The size $|A|$ of a formula A is the number of occurrences of atoms and constants in A.

2.2 Proof systems

We denote by FORM the set of propositional formulae, by TAUT the set of tautologies, and by SAT the set of satisfiable formulae.

In what follows we call a partial function f a *polynomial-time partial function* if it is polynomial-time computable on inputs on which it is defined and if the set of all such inputs, denoted $\mathsf{dom}(f)$, is polynomial-time decidable.

Definition 2.1 (Abstract proof systems)**.** A *propositional proof system* (PPS) P over an alphabet Σ is a polynomial-time partial function $\Sigma^* \to \mathsf{FORM}$.

If $P(\pi) = A$ then we say that π is a P-proof with conclusion A, or just a P-proof of A. If $\mathsf{ran}(P) \subseteq \mathsf{TAUT}$ then we say that P is *sound*, and if $\mathsf{ran}(P) \supseteq \mathsf{TAUT}$ then we say that P is *complete*.

The size $|\pi|$ of a proof π is its length. If P and Q are PPSs then we say that P *polynomially simulates* Q if there is a polynomial-time function $\mathsf{dom}(Q) \to \mathsf{dom}(P)$ such that, whenever $\pi \mapsto \pi'$ and $Q(\pi) = \tau$, we have $P(\pi) = \tau$. If P and Q polynomially simulate each other we say that they are *polynomially equivalent*.

The definition we give above of an abstract proof system differs slightly from others appearing in the literature, e.g. in [CR74b]. Namely, we regard proof systems as partial functions defined only on those strings that code a real proof rather than fully defined functions. In the latter case this is dealt with by simply mapping such strings to $\top$. It is not difficult to see that the two definitions are equivalent, but we prefer ours since it is closer to intuition, as illustrated by the example below.

Example 2.2 (Truth tables)**.** Let $\mathsf{tt}(A)$ denote the truth table generated by a formula A, encoded as just the concatenation of its columns, under some fixed convention on the order columns should appear. For example, here is the truth table for the formula $[(a \wedge b) \vee c]$,

a	b	c	$(a \wedge b)$	$[(a \wedge b) \vee c]$
0	0	0	0	0
0	0	1	0	1
0	1	0	0	0
0	1	1	0	1
1	0	0	0	0
1	0	1	0	1
1	1	0	1	1
1	1	1	1	1

and here is its encoding as a string:[1]

$$a00001111b00110011c01010101(a \wedge b)00000011[(a \wedge b) \vee c]01010111$$

[1]Strictly speaking, we should also fix a (prefix-free) encoding of propositional formulae over some finite alphabet, but this is routine so we omit this consideration in the example.

This induces a sound and complete PPS T by $T^{-1}(\tau) = \{\mathsf{tt}(\tau)\}$ for $\tau \in \mathsf{TAUT}$. Clearly we can check in polynomial time whether a string encodes a valid truth table, and we can check if the table is a proof by checking that the final column has 1 in every entry, except of course the header.

T is not a particularly interesting proof system: it has only one proof of each tautology, and this can be generated in a uniform way, so there is no creativity in the design of a proof.

When we define actual proof systems, we will omit the formalities of encodings etc. It should be clear that a formal PPS can be recovered from the given definitions, like in the above example for truth tables.

All of the proofs this dissertation are constructive and so, while we typically speak of the 'existence' of proofs (or other related objects) of a certain size (or other appropriate measure), it should be understood that they can be constructed in time polynomial in their size. This is consistent with standard conventions in the proof complexity literature.

2.3 Connections to computational complexity

While we do not address questions of complexity theory in this work, it will be helpful to put into context some of the motivations behind the study of propositional proof complexity. In a sequence of results from the late '60s and early '70s, Cook and others found deep connections between propositional proofs and complexity theory, and we restate them here with informal proofs.

The starting point of this programme of research is the Cook-Levin theorem, which first appeared in [Coo71]:

Theorem 2.3 (Cook-Levin). SAT *is* NP*-complete.*

Corollary 2.4. TAUT *is* coNP*-complete.*

Proof. Since SAT is NP-complete we have that SAT^c is coNP-complete. Clearly we can check whether a string is a formula in polynomial time and so we have that the set UNSAT of unsatisfiable formulae is coNP-complete. Finally the map $A \mapsto \bar{A}$ is a linear-time bijection on FORM whose restriction to UNSAT is a bijection to TAUT, whence the result follows. □

From here we gain our first connection to computational complexity: deciding whether a formula is a tautology or not can be done in polynomial time if and only if $\mathsf{P} = \mathsf{NP}$, since P is closed under complements:

Corollary 2.5. $\mathsf{TAUT} \in \mathsf{P}$ *if and only if* $\mathsf{P} = \mathsf{NP}$.

A more subtle connection, due to Cook and Reckhow [CR74b], relates the size of proofs to the relationship between nondeterministic classes and their complements.

Definition 2.6. We call a PPS P *super* if it is sound and complete and, for every $\tau \in \mathsf{TAUT}$, there is some P-proof π of τ of size polynomial in $|\tau|$.

Theorem 2.7 (Cook-Reckhow)**.** $\mathsf{NP} = \mathsf{coNP}$ *if and only if there exists a super PPS.*

Proof. Suppose $\mathsf{NP} = \mathsf{coNP}$. By Cook-Levin we then have that TAUT is in NP, so let M be a nondeterministic Turing machine accepting TAUT in polynomial time. Let R_τ be the set of accepting runs of M on input τ, construed over some alphabet Σ, and define a PPS $P : \Sigma^* \to \mathsf{TAUT}$ by $P^{-1}(\tau) = R_\tau$. Then, since M accepts τ in time polynomial in $|\tau|$, there must be some polynomial-length run π of M on input τ, i.e. π is a polynomial-size P-proof of τ.

Conversely, suppose P is a super PPS over an alphabet Σ with bounding polynomial p. Since TAUT is coNP-complete, by Cook-Levin, it suffices to show that $\mathsf{TAUT} \in \mathsf{NP}$. We define the following nondeterministic algorithm:

```
input : $A \in \mathsf{FORM}$
Let $w = \epsilon$
for $i = 0$ to $p(|A|)$ do
    (1) Check $P(w) = A$. If output is yes then output: yes
    (2) Choose a letter $a \in \Sigma$ and let $w = wa$
end
output: no
```

If A is a tautology then there must be some word $\pi \in \Sigma^*$ with $P(\pi) = A$ and $|\pi| \leq p(|A|)$, and so there will be some sequence of choices (at step (2)) that ends on such a π. The length of π is bounded by $p(|A|)$ and we invoke P $|\pi|$-many times in the process, so the algorithm accepts A in time polynomial in $|A|$, as required. □

2.4 Some remarks

The final result above has inspired what is called *Cook's programme* [Bus11] for separating NP and coNP: if we can prove that all PPSs have superpolynomial lower bounds then $\mathsf{NP} \neq \mathsf{coNP}$ will follow. So far this has proved (unsurprisingly) a difficult task; while lower bounds were readily proved for comparatively weak PPSs, there seems to be a barrier at the level of Hilbert-Frege systems, or equivalently sequent calculi with cut, where only small-degree polynomial lower bounds exist.

To this end researchers have attempted to gradually bridge the gap between Hilbert-Frege systems and ones for which we have nontrivial lower bounds, e.g. by restricting what cuts in a proof are permitted. It is in this direction, that this dissertation makes some small contribution towards complexity theory, with the system KS being of particular interest.

Chapter 3

Deep inference proof theory

In this chapter we present the basics of deep inference proof theory that underline this work. It should be clear how the systems we define can be made to fit into the abstract proof complexity setting outlined in the previous chapter, so we omit formal encodings.

More comprehensive overviews can be found e.g. in [Brü04] from a structural proof theory point of view or in [BG09b] from a proof complexity point of view. In any case all the notions in this chapter are standard in the deep inference literature.

A *context* is a formula with a hole occurring in place of a subformula. Formally they are generated by the following grammar:

$$\xi\{\ \} \quad ::= \quad \{\ \} \quad | \quad A \wedge \xi\{\ \} \quad | \quad A \vee \xi\{\ \} \quad | \quad \xi\{\ \} \wedge A \quad | \quad \xi\{\ \} \vee A$$

We write $\xi\{A\}$ to denote the formula obtained by substituting the formula A for the hole $\{\ \}$ in $\xi\{\ \}$.

Terms are freely built from the language of propositional logic together with formula variables A, B etc. and the duality symbol $\bar{\cdot}$. A *renaming* is function on the set of atoms and an *instance* of a term t is the formula obtained by some substitution of formulae for formula variables and some renaming of atoms in t, interpreting the duality symbol $\bar{\cdot}$ on a formula substituted for a formula variable as explained in Sect. 2.1.

We often use square brackets, $[,]$, for disjunctions and round brackets, $(,)$, for conjunctions, to ease parsing of formulae, contexts, terms etc.

3.1 Proof systems and derivations

Definition 3.1 (Inference rules and systems)**.** An inference rule ρ is an expression $\rho\,\dfrac{s}{t}$ for some terms s and t. An *instance* of ρ is obtained from $\rho\,\dfrac{s}{t}$ by replacing s and t by instances, respectively, determined by the same renaming of atoms and substitutions of formulae for formula variables. We call ρ *sound* if, for any instance $\rho\,\dfrac{A}{B}$, A logically implies B.

A *system* is a finite set of inference rules.

Definition 3.2 (Equality)**.** We define twelve inference rules below, which we collectively refer to as $=$, as an abuse of notation.

$$
\begin{array}{cc@{\qquad}cccc}
\multicolumn{2}{c}{\text{Rebracketing rules}} & \multicolumn{4}{c}{\text{Unit rules}} \\[1ex]
=\dfrac{A \vee B}{B \vee A} & =\dfrac{[A \vee B] \vee C}{A \vee [B \vee C]} & =\dfrac{A}{A \wedge \top} & =\dfrac{A \wedge \top}{A} & =\dfrac{\bot}{\bot \wedge \bot} & =\dfrac{\bot \wedge \bot}{\bot} \\[2ex]
=\dfrac{A \wedge B}{B \wedge A} & =\dfrac{(A \wedge B) \wedge C}{A \wedge (B \wedge C)} & =\dfrac{A \vee \bot}{A} & =\dfrac{A}{A \vee \bot} & =\dfrac{\top \vee \top}{\top} & =\dfrac{\top}{\top \vee \top} \\[1ex]
\textit{commutativity} & \textit{associativity} & & & &
\end{array}
$$

We implicitly assume that these rules are contained in every deep inference system.

Definition 3.3 (Derivations)**.** Derivations, denoted Φ, Ψ etc., along with premiss and conclusion functions (pr, cn respectively) are defined as follows:

1. Every formula A is a derivation with premiss and conclusion A.
2. If Φ and Ψ are derivations then $(\Phi \star \Psi)$ is a derivation for $\star \in \{\wedge, \vee\}$ with premiss $(\mathsf{pr}(\Phi) \star \mathsf{pr}(\Psi))$ and conclusion $(\mathsf{cn}(\Phi) \star \mathsf{cn}(\Psi))$.
3. If Φ and Ψ are derivations and $\rho\,\dfrac{\mathsf{cn}(\Phi)}{\mathsf{pr}(\Psi)}$ is an instance of an inference rule ρ then $\rho\,\dfrac{\Phi}{\Psi}$ is a derivation with premiss $\mathsf{pr}(\Phi)$ and conclusion $\mathsf{cn}(\Psi)$.

If $\mathsf{pr}(\Phi) \equiv \top$ then we say that Φ is a *proof.* If every inference step in a derivation Φ is an instance of a rule in a system $\mathcal{S}$ then we say Φ is an $\mathcal{S}$-derivation, and if $\mathsf{pr}(\Phi) \equiv \top$ then Φ is an $\mathcal{S}$-proof.

We write $\begin{array}{c} A \\ \Phi \| \mathcal{S} \\ B \end{array}$ to denote an $\mathcal{S}$-derivation Φ from A to B, i.e. with premiss A and conclusion B, and we write $\begin{array}{c} \Phi \overline{\|} \mathcal{S} \\ A \end{array}$ to denote an $\mathcal{S}$-proof of A.

We call a system $\mathcal{S}$ *sound* if, for every derivation $\begin{array}{c} A \\ \| \mathcal{S} \\ B \end{array}$, A logically implies B, and we call $\mathcal{S}$ *complete* if, for every tautology τ, there is a $\mathcal{S}$-proof of τ.

Sometimes we write derivations with formula variables occurring, for example when showing that some rule is derivable; these should be understood as templates for any derivation arising from instantiating the formula variables by formulae.

Definition 3.4. We define the deep inference systems SKS and KS below.

$$
\mathsf{SKS}\left\{
\begin{array}{cccc}
\multicolumn{3}{c}{\text{Atomic structural rules}} & \text{Linear logical rules} \\[2ex]
\mathsf{ai}\!\uparrow \dfrac{a \wedge \bar{a}}{\bot} & \mathsf{aw}\!\uparrow \dfrac{a}{\top} & \mathsf{ac}\!\uparrow \dfrac{a}{a \wedge a} & \mathsf{m}\,\dfrac{(A \wedge B) \vee (C \wedge D)}{[A \vee C] \wedge [B \vee D]} \\
\textit{cut} & \textit{coweakening} & \textit{cocontraction} & \textit{medial} \\[2ex]
\mathsf{ai}\!\downarrow \dfrac{\top}{a \vee \bar{a}} & \mathsf{aw}\!\downarrow \dfrac{\bot}{a} & \mathsf{ac}\!\downarrow \dfrac{a \vee a}{a} & \mathsf{s}\,\dfrac{A \wedge [B \vee C]}{(A \wedge B) \vee C} \\
\textit{identity} & \textit{weakening} & \textit{contraction} & \textit{switch}
\end{array}
\right.
\quad \left.\begin{array}{c} \\ \\ \\ \end{array}\right\}\mathsf{KS}
$$

We also define the system $\mathsf{KS}^+ = \mathsf{SKS} \setminus \{\mathsf{ai}\!\uparrow\}$. Recall that the $=$-rules are implicitly contained in all these systems. Importantly, note the distinction between variables for formulae and atoms in the rules above.

By observing the soundness of each rule, notice that SKS and all its subsystems are sound. According to [Brü04] SKS stands for 'symmetric klassisch system'.

Example 3.5. We give an example of an KS^+-proof of $(a \wedge a) \vee \bar{a}$ below:

$$
=\,\dfrac{\mathsf{ai}\!\downarrow\dfrac{\top}{a \vee \left(\mathsf{s}\,\dfrac{=\,\dfrac{\bar a}{\mathsf{ai}\!\downarrow\dfrac{\top}{a \vee \bar a} \wedge \bar a}}{a \vee (\bar a \wedge \bar a)}\right)}}{\mathsf{ac}\!\uparrow\dfrac{\mathsf{ac}\!\downarrow\dfrac{a \vee a}{a}}{a \wedge a} \vee \mathsf{m}\,\dfrac{(\bar a \wedge \bar a) \vee \left(\bot \wedge \mathsf{aw}\!\downarrow\dfrac{\bot}{\bar a}\right)}{=\,\dfrac{=\,\dfrac{\bar a \vee \bot}{\bar a} \wedge \mathsf{ac}\!\downarrow\dfrac{\bar a \vee \bar a}{\mathsf{aw}\!\uparrow\dfrac{\bar a}{\top}}}{\bar a}}}
$$

Notice that, at the s-step in the middle of the derivation above, we have technically omitted two commutativity steps, one before and one after, since the disjunction is on the right. Clearly this does not cause any problem in parsing the proof, and in fact aids legibility, so we will often similarly omit $=$-steps when it is helpful to do so. In the same spirit, when counting the number of inferences in a derivation, we might implicitly ignore $=$-steps if it is helpful to do so in, say, an induction argument

Other notational conventions we adopt are that we write $\rho \overset{A}{\underset{B}{=}}$ if there is a derivation $\begin{array}{c} A \\ \Vert\{\rho\} \\ B \end{array}$ and $n{\cdot}\rho \dfrac{A}{B}$ if there is a derivation $\begin{array}{c} A \\ \Vert\{\rho\} \\ B \end{array}$ with n ρ-steps occurring.

When manipulating derivations we use the terms 'descendant' and 'ancestor' in the natural way: in an inference step that is an instance of a rule $\rho\,\dfrac{s}{t}$, any atom occurrence in the instance of t is a descendant of just those occurrences of the same atom in the instance of s that are renamings of the same atom symbol, or are in the same position in an instance of the same formula variable. In a derivation we close the notion of descendant under transitivity and use 'ancestor' as the inverse notion of descendant.

It is not difficult to see that $\mathsf{aw}\!\uparrow$-steps in an KS^+-proof can be eliminated in linear time, and so KS^+ is polynomially simulated by its subsystem $\mathsf{KS} \cup \{\mathsf{ac}\!\uparrow\}$. This is an easy corollary of the results in Chapt. 6, namely Prop. 6.14, but we sketch a direct proof here to give a flavour of the kind of arguments commonplace in deep inference proof theory.

Proposition 3.6. $\mathsf{KS} \cup \{\mathsf{ac}\!\uparrow\}$ *linearly simulates* KS^+*.*

Proof. In a KS^+-derivation Φ, replace every $\mathsf{aw}\!\uparrow$-instance $\frac{a}{\top}$ and every ancestor of a by $\top$, until an $\mathsf{aw}\!\downarrow$, $\mathsf{ac}\!\uparrow$ or $\mathsf{ai}\!\downarrow$ step is reached. Any logical steps remain valid, and structural steps are altered as follows,

$$\mathsf{ac}\!\downarrow \frac{a \vee a}{a} \quad \rightarrow \quad = \frac{\top \vee \top}{\top}$$

$$\mathsf{aw}\!\downarrow \frac{\bot}{a} \quad \rightarrow \quad \begin{array}{c} \bot \\ \bullet \| \\ \top \end{array}$$

$$\mathsf{ac}\!\uparrow \frac{a}{a \wedge a} \quad \rightarrow \quad = \frac{a}{\top \wedge a}$$

$$\mathsf{ai}\!\downarrow \frac{\top}{a \vee \bar{a}} \quad \rightarrow \quad = \frac{\top}{\top \vee \mathsf{aw}\!\downarrow \frac{\bot}{\bar{a}}}$$

where the derivation marked $\bullet$ is obtained for $\{\mathsf{s}\}$ or $\{\mathsf{m}\}$ by Rmk. 3.15. $\square$

Notice that, since all the alterations to structural steps are local, this $\mathsf{aw}\!\uparrow$-elimination procedure in fact holds for any subsystem of SKS that contains s or m whenever it contains $\mathsf{aw}\!\downarrow$, and further contains $\mathsf{aw}\!\downarrow$ whenever it contains $\mathsf{ai}\!\downarrow$. In particular we have the following result:

Corollary 3.7. KS *linearly simulates* $\mathsf{KS} \cup \{\mathsf{aw}\!\uparrow\}$*.*

3.2 Duality

Due to the lack of distinction between object and meta levels in deep inference, proofs and derivations have many more symmetries than in other systems. In particular, there is an elegant top-down symmetry induced by the De Morgan laws in deep inference.

Definition 3.8 (Dual Systems)**.** The dual of a rule $\rho \frac{s}{t}$ is $\bar{\rho} \frac{\bar{t}}{\bar{s}}$. The set of duals of a system $\mathcal{S}$ is denoted $\bar{\mathcal{S}}$.

For example in SKS a structural rule $\rho\downarrow$ is dual to $\rho\uparrow$, while switch and medial are self-dual.[1] Notice that a rule is sound if and only if its dual is by the law of contraposition. This idea can be extended to whole proofs:

Proposition 3.9. *We can transform derivations* $\begin{array}{c}\bar{B}\\ \Vert \bar{\mathcal{S}} \\ \bar{A}\end{array}$ *into derivations* $\begin{array}{c}A\\ \Vert \mathcal{S} \\ B\end{array}$ *in linear time.*

Proof. Flip the derivation upside-down and replace every rule with its dual. □

Corollary 3.10. *A system is complete if and only if its dual is refutationally complete, i.e. can derive* $\bot$ *from every unsatisfiable formula.*

We give an example now of a result that we appeal to throughout this work whose proof is made simpler by appealing to duality.

Lemma 3.11. *There are derivations* $\begin{array}{c}\xi\{A\}\\ \Vert \{\mathsf{s}\} \\ A \vee \xi\{\bot\}\end{array}$ *and* $\begin{array}{c}A \wedge \xi\{\top\}\\ \Vert \{\mathsf{s}\} \\ \xi\{A\}\end{array}$ *of size* $O(|\xi\{A\}|^2)$*.*

Proof. We construct only the first derivation, with the second following by duality due to Prop. 3.9. We proceed by induction on the depth of the hole in ξ. The base case is trivial, $= \frac{A}{A \vee \bot}$, and we give the inductive steps below,

$$\mathsf{s}\,\frac{B \wedge \begin{array}{c}\xi\{A\}\\ {\scriptstyle IH}\Big\Vert \{\mathsf{s}\} \\ A \vee \xi\{\bot\}\end{array}}{A \vee (B \wedge \xi\{\bot\})} \quad , \qquad =\frac{B \vee \begin{array}{c}\xi\{A\}\\ {\scriptstyle IH}\Big\Vert \{\mathsf{s}\} \\ A \vee \xi\{\bot\}\end{array}}{A \vee [B \vee \xi\{\bot\}]}$$

where derivations marked IH are obtained by the inductive hypothesis. □

3.3 Alternative forms of proofs

It is often useful to consider alternative but equivalent systems and forms of proof to conduct proof theoretic arguments. In this section we define generic rules, from which completeness of the systems so far defined is easily obtained, and also the *Calculus of Structures* (CoS) form of proofs, which admits induction measures suitable for many of the arguments in Part II.

[1] Strictly speaking, since rules are defined as expressions, they are only dual modulo the relative use of the duality symbol $\bar{\cdot}$, but we generally ignore this technicality.

3.3.1 Generic rules and systems

While the systems we have so far defined have atomic structural rules, we could equivalently consider 'generic' versions of these rules and define associated systems. Observing that both approaches are polynomially equivalent, many arguments about SKS can be simplified, as we will see below.

Definition 3.12. We define the systems SKSg and KSg below:

$$\begin{array}{rcccc}
 & & \text{Structural rules} & & \text{Logical rule} \\
\mathsf{SKSg}\left\{\begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \end{array}\right. &
\begin{array}{c}
\mathsf{i}{\uparrow}\dfrac{A\wedge\bar{A}}{\bot} \\ \textit{cut} \\[2ex] \mathsf{i}{\downarrow}\dfrac{\top}{A\vee\bar{A}} \\ \textit{identity}
\end{array} &
\begin{array}{c}
\mathsf{w}{\uparrow}\dfrac{A}{\top} \\ \textit{coweakening} \\[2ex] \mathsf{w}{\downarrow}\dfrac{\bot}{A} \\ \textit{weakening}
\end{array} &
\begin{array}{c}
\mathsf{c}{\uparrow}\dfrac{A}{A\wedge A} \\ \textit{cocontraction} \\[2ex] \mathsf{c}{\downarrow}\dfrac{A\vee A}{A} \\ \textit{contraction}
\end{array} &
\begin{array}{c}
 \\ \\[2ex] \left.\begin{array}{c}\mathsf{s}\dfrac{A\wedge[B\vee C]}{(A\wedge B)\vee C} \\ \textit{switch}\end{array}\right\}\mathsf{KSg}
\end{array}
\end{array}$$

Proposition 3.13. SKSg *is polynomially equivalent to* SKS *and* KSg *is polynomially equivalent to* KS*.*

Proof. Since each atomic step is a special case of a generic step, to show that the generic systems polynomially simulate their respective local systems it suffices to derive medial in KSg:

$$\mathsf{m}\dfrac{(A\wedge B)\vee(C\wedge D)}{[A\vee C]\wedge[B\vee D]} \quad\to\quad \mathsf{c}{\downarrow}\dfrac{\left(=\dfrac{A}{A\vee\mathsf{w}{\downarrow}\dfrac{\bot}{C}}\wedge=\dfrac{B}{B\vee\mathsf{w}{\downarrow}\dfrac{\bot}{D}}\right)\vee\left(=\dfrac{C}{\mathsf{w}{\downarrow}\dfrac{\bot}{A}\vee C}\wedge=\dfrac{D}{\mathsf{w}{\downarrow}\dfrac{\bot}{B}\vee D}\right)}{[A\vee C]\wedge[B\vee D]}$$

Notice that we have only used the rules $\mathsf{w}{\downarrow}$ and $\mathsf{c}{\downarrow}$ above. Since medial is self-dual, we could have also derived it using $\mathsf{w}{\uparrow}$ and $\mathsf{c}{\uparrow}$, by Prop. 3.9.

In the other direction it suffices to show that each atomic rule can, along with switch and medial, derive its respective generic rule. We argue by structural induction on the formulae occurring in a generic step.

$$\mathsf{i}{\downarrow}\frac{\top}{(A\wedge B)\vee\bar{A}\vee\bar{B}} \quad\to\quad {=}\frac{\top}{2{\cdot}\mathsf{s}\dfrac{\left(\begin{array}{c}\top\\ IH\Big\Vert\{\mathsf{ai}{\downarrow},\mathsf{s}\}\\ A\vee\bar{A}\end{array}\wedge\begin{array}{c}\top\\ IH\Big\Vert\{\mathsf{ai}{\downarrow},\mathsf{s}\}\\ B\vee\bar{B}\end{array}\right)}{(A\wedge B)\vee\bar{A}\vee\bar{B}}}$$

$$\mathsf{w}{\downarrow}\frac{\bot}{A\star B} \quad\to\quad {=}\frac{\bot}{\left(\begin{array}{c}\bot\\ IH\Big\Vert\{\mathsf{aw}{\downarrow},\mathsf{s}\}\\ A\end{array}\star\begin{array}{c}\bot\\ IH\Big\Vert\{\mathsf{aw}{\downarrow},\mathsf{s}\}\\ B\end{array}\right)}$$

$$\mathsf{c}{\downarrow}\frac{[A\vee B]\vee[A\vee B]}{A\vee B} \quad\to\quad {=}\frac{[A\vee B]\vee[A\vee B]}{\left[\begin{array}{c}A\vee A\\ IH\Big\Vert\{\mathsf{ac}{\downarrow},\mathsf{m}\}\\ A\end{array}\vee\begin{array}{c}B\vee B\\ IH\Big\Vert\{\mathsf{ac}{\downarrow},\mathsf{m}\}\\ B\end{array}\right]}$$

$$\mathsf{c}{\downarrow}\frac{(A\wedge B)\vee(A\wedge B)}{A\wedge B} \quad\to\quad \mathsf{m}\frac{(A\wedge B)\vee(A\wedge B)}{\left(\begin{array}{c}A\vee A\\ IH\Big\Vert\{\mathsf{ac}{\downarrow},\mathsf{m}\}\\ A\end{array}\wedge\begin{array}{c}B\vee B\\ IH\Big\Vert\{\mathsf{ac}{\downarrow},\mathsf{m}\}\\ B\end{array}\right)}$$

where $\star\in\{\wedge,\vee\}$ and derivations marked IH are obtained by the inductive hypothesis. The derivations for $\uparrow$ rules are obtained by duality. $\square$

Remark 3.14. Notice that the above result can be generalised to all subsystems $\mathcal{S}$ of SKS and their generic analogues, provided $\mathcal{S}\supseteq\{\mathsf{ac}{\downarrow},\mathsf{aw}{\downarrow},\mathsf{s},\mathsf{m}\}$, or dually $\mathcal{S}\supseteq\{\mathsf{ac}{\uparrow},\mathsf{aw}{\uparrow},\mathsf{s},\mathsf{m}\}$. For example, $\mathsf{KSg}\cup\{\mathsf{c}{\uparrow},\mathsf{w}{\uparrow}\}$ is polynomially equivalent to KS^{+}, which in turn is equivalent to $\mathsf{KS}\cup\{\mathsf{ac}{\uparrow}\}$ by Prop. 3.6 and so also $\mathsf{KSg}\cup\{\mathsf{c}{\uparrow}\}$.

Remark 3.15. We have technically omitted the special case when an instance of $\mathsf{w}{\downarrow}$ or $\mathsf{w}{\uparrow}$ is $\frac{\bot}{\top}$. We cannot strictly mimic this case by an $\mathsf{aw}{\downarrow}$ or $\mathsf{aw}{\uparrow}$ step since renamings do not map atoms to units. In fact $\dfrac{\bot}{\top}$ is derivable for $\{\mathsf{s}\}$ and $\{\mathsf{m}\}$,

$$
{=}\dfrac{{=}\dfrac{\bot}{\mathsf{m}\dfrac{{=}\dfrac{\bot}{\bot\wedge\top}\vee{=}\dfrac{\bot}{\bot\wedge\top}}{{=}\dfrac{\bot\vee\top}{\top}\wedge{=}\dfrac{\bot\vee\top}{\top}}}}{\top}
\quad,\qquad
{=}\dfrac{{=}\dfrac{\bot}{\mathsf{s}\dfrac{\bot\wedge{=}\dfrac{\top}{\bot\vee\top}}{{=}\dfrac{\bot\wedge\bot}{\bot}\vee\top}}}{\top}
$$

so such steps can be simulated by one of the derivations above.

Perhaps the most useful reason for considering these generic systems is that they can make translations to and from other systems simpler and more intuitive, allowing us to obtain some basic proof complexity results almost by inspection.

The following theorem first appeared in [BG09b], to which we refer the reader for a fully detailed proof; notice that the first half of the proof is simplified by appealing to generic systems.

Theorem 3.16. SKS *is polynomially equivalent to Hilbert-Frege systems.*

Proof sketch. Notice that the rules of $\mathsf{KSg}\cup\{\mathsf{i}\!\uparrow\}$, along with our mechanisms for proof composition, are simply a generalisation of a one-sided sequent calculus with cut, which are polynomially equivalent to Hilbert-Frege systems. Since SKS polynomially simulates SKSg by Prop. 3.13 and $\mathsf{SKSg}\supseteq\mathsf{KSg}\cup\{\mathsf{i}\!\uparrow\}$, we have that SKS polynomially simulates Hilbert-Frege systems.

In the other direction we appeal to the robustness of Hilbert-Frege: all Hilbert-Frege systems are polynomially equivalent [CR74a]. We design a system with axioms $s\supset t$ for each inference rule $\rho\dfrac{s}{t}$ of SKSg, $A\supset A$ and the following rules:

$$
\top\,\dfrac{\top\supset A}{A}\qquad t\,\dfrac{A\supset B\quad B\supset C}{A\supset C}\qquad \wedge\,\dfrac{A\supset B\quad C\supset D}{(A\wedge C)\supset(B\wedge D)}\qquad \vee\,\dfrac{A\supset B\quad C\supset D}{(A\vee C)\supset(B\vee D)}
$$

Now we define a translation H from SKSg-derivations $\begin{array}{c}A\\ \Phi\Vert\mathsf{SKSg}\\ B\end{array}$ to Hilbert-Frege proofs

$H\Phi$ of $A \supset B$ as follows,

$$A \quad \to \quad \dfrac{}{A \supset A}$$

$$\begin{array}{c} A \\ \Psi \Big\| \\ B \end{array} \star \begin{array}{c} C \\ \Theta \Big\| \\ D \end{array} \quad \to \quad \star\,\dfrac{\begin{array}{c} H\Psi \Big\| \\ A \supset B \end{array} \quad \begin{array}{c} H\Theta \Big\| \\ C \supset D \end{array}}{(A \star C) \supset (B \star D)}$$

$$\begin{array}{c} A \\ \Psi \| \\ B \\ \rho\,\dfrac{}{} \\ B' \\ \Theta \| \\ C \end{array} \quad \to \quad t\,\dfrac{t\,\dfrac{\begin{array}{c} H\Psi \| \\ A \supset B \end{array} \quad \rho\,\dfrac{}{B \supset B'}}{A \supset B'} \quad \begin{array}{c} H\Theta \Big\| \\ B' \supset C \end{array}}{A \supset C}$$

for $\star \in \{\wedge, \vee\}$. Finally the image of an SKSg-proof of a formula A under H is $\top \supset A$, whence A can be derived by an application of $\top$. $\square$

Notice that the second half of the above argument was not at all dependent on the choice of rules of SKSg, and so can in fact be generalised to arbitrary systems:

Corollary 3.17. *Hilbert-Frege systems polynomially simulate every deep inference system.*

Notice also that, by the first half of the above argument, we can now deduce the completeness property.

Corollary 3.18. SKS *is sound and complete.*

The completeness of systems between KS and SKS can be shown in the same way, although only the simulation in one direction holds: KSg polynomially simulates cut-free sequent calculi but not vice-versa, as we will see in Chapt. 8. In any case, the results of Chapt. 5 and Chapt. 6 outline explicit normalisation procedures from SKS-proofs to KS^+ and KS proofs.

3.3.2 Calculus of Structures

The Calculus of Structures (CoS) was the first deep inference formalism defined for propositional logic [BT01] [BG09b]. Derivations are construed as just a sequence of formulae with each formula following from the previous one by application of some inference rule inside some context.

This notion of derivation is polynomially equivalent to the notion given in Dfn. 3.3, which we refer to simply as '(deep inference) derivations', although it is arguably clearer from the CoS formalism why this area of proof theory is called *deep* inference.

Definition 3.19. CoS derivations Φ with premiss A and conclusion B, denoted $\begin{array}{c} A \\ \Phi \| \\ B \end{array}$, are defined as follows:

1. Every formula A is a CoS derivation with premiss and conclusion itself.
2. If $\begin{array}{c} A \\ \Phi \| \\ \xi\{B\} \end{array}$ is a CoS derivation and $\rho \frac{B}{C}$ is an instance of a rule ρ then $\begin{array}{c} A \\ \Phi \| \\ \rho \dfrac{\xi\{B\}}{\xi\{C\}} \end{array}$ is a CoS derivation.

CoS proofs and CoS derivations in a system $\mathcal{S}$, along with their notations, are defined in the same way as for deep inference derivations in Dfn. 3.3.

We point out that in [GGP10] Guglielmi et al. define a formalism, *open deduction*, that generalises both CoS and deep inference derivations. In that work CoS derivations are called *sequential*, while the derivations we previously defined are called *synchronal*.

We sketch a proof of the following proposition, that both forms are polynomially equivalent, but refer the reader to [GGP10] for full details of the proof.

Proposition 3.20. *For any system $\mathcal{S}$, CoS $\mathcal{S}$-derivations can be polynomially transformed into deep inference $\mathcal{S}$-derivations, preserving premiss and conclusion, and vice-versa.*

Proof sketch. See, e.g. [GGP10] for full details. We define a transformation T from CoS derivations to deep inference derivations by induction on the length of a CoS derivation:

$$A \quad \rightarrow \quad A$$

$$\begin{array}{c} A \\ \Phi \Big\| \mathcal{S} \\ \rho \dfrac{\xi\{B\}}{\xi\{C\}} \end{array} \quad \rightarrow \quad \begin{array}{c} A \\ T\Phi \Big\| \mathcal{S} \\ \xi \left\{ \rho \dfrac{B}{C} \right\} \end{array}$$

In the other direction we define a transformation U from deep inference derivations to

CoS derivations by induction on the structure of a deep inference derivation,

$$A \quad \to \quad A$$

$$\begin{array}{c} A \\ {\scriptstyle \Phi}\Big\| {\scriptstyle \mathcal{S}} \\ B \end{array} \star \begin{array}{c} C \\ {\scriptstyle \Phi}\Big\| {\scriptstyle \mathcal{S}} \\ D \end{array} \quad \to \quad \begin{array}{c} A \star C \\ {\scriptstyle U\Phi \star C}\Big\| {\scriptstyle \mathcal{S}} \\ B \star C \\ {\scriptstyle B \star U\Psi}\Big\| {\scriptstyle \mathcal{S}} \\ B \star D \end{array}$$

$$\begin{array}{c} A \\ {\scriptstyle \Phi}\Big\| {\scriptstyle \mathcal{S}} \\ {\scriptstyle \rho}\,\dfrac{B}{C} \end{array} \quad \to \quad \begin{array}{c} A \\ {\scriptstyle U\Phi}\Big\| {\scriptstyle \mathcal{S}} \\ {\scriptstyle \rho}\,\dfrac{B}{C} \end{array}$$

for $\star \in \{\wedge, \vee\}$, where $U\Phi \star C$ is obtained by replacing each line X in $U\Phi$ by $X \star C$, and $B \star U\Psi$ similarly.

T clearly has complexity linear in the size of an input derivation whereas U has quadratic complexity, since it essentially just 'completes the square' of a deep inference derivation. $\square$

Notice that the translations U and T above preserve the number of inference steps in a derivation, and so deep inference derivations inherit a notion of *length* from their associated CoS derivations. This provides us with an induction measure that proves useful in later sections.

Part II

Complexity in the presence of cocontraction

Chapter 4

Bounded deep inference

Our current notion of derivation allows for rules to operate arbitrarily deep, in the sense that inference steps can occur under the scope of unboundedly many alternations of $\wedge$ and $\vee$. Indeed, one might claim that it is this very flexibility that allows for the quasipolynomial-time normalisation procedure that we will see in the next chapter. While this is somewhat true, the results of this chapter show that proofs of the same size, up to a polynomial, can be constructed even when the *depth* of inference steps is bounded. This refutes a conjecture of Bruscoli and Guglielmi that no bounded analytic deep inference system can simulate its unbounded counterpart [BG09b].

The material in this chapter is based on work that has appeared in [Das11].

It will be convenient to consider proofs and derivations in CoS form in this chapter, and we also work with generic rules to retain completeness when the depth of inference is bounded.

Definition 4.1 (Depth)**.** For a formula A its depth, $\delta(A)$, is the maximum number of alternations of $\wedge$ and $\vee$ in its formula tree. For a context $\xi\{\ \}$, the depth of its hole, $\delta(\{\ \}, \xi\{\ \})$, is the number of alternations in the path to the hole in the context's formula tree. The depth of a subformula is defined similarly.

The depth of an inference step is the depth of the hole in which the instance of the associated inference rule occurs.

When calculating depth we adopt the convention that every formula or context has an outer $\wedge$. For example:

$$\delta(a \vee (b \wedge c)) = \delta(\{\ \}, a \vee (b \wedge \{\ \})) = 2 \qquad \delta(a \wedge (b \wedge (c \wedge d))) = 0$$

Notation 4.2. For a system $\mathcal{S}$ we write k-$\mathcal{S}$ to denote the system whose derivations are just $\mathcal{S}$-derivations where all inference steps have depth less than or equal to k.

For an inference step ρ, we will sometimes indicate its depth in parentheses on the right, e.g. $\rho(3)\,\dfrac{A}{B}$ indicates that $\delta(\rho) = 3$, and we write $\mathcal{S} \cup \{\rho(k)\}$ to denote the system whose derivations are just $\mathcal{S} \cup \{\rho\}$-derivations with all ρ steps having depth k.

For a context $\xi\{\ \}$, the depth of its hole may be indicated as a superscript, *e.g.* $\xi^2\{\ \}$ for a context with a hole at depth 2.

4.1 The depth-change trick

In this section we show that, in the presence of cocontraction, deep inference derivations can be polynomially transformed into bounded deep inference derivations preserving premiss and conclusion.

Definition 4.3. We define $\mathsf{aSKSg} = \{\mathsf{i}\downarrow, \mathsf{w}\downarrow, \mathsf{c}\downarrow, \mathsf{c}\uparrow, \mathsf{s}, \mathsf{m}\}$, i.e. the set of all the usual analytic rules of deep inference (in particular there is no $\mathsf{i}\uparrow$ and no $\mathsf{w}\uparrow$), with generic versions of the structural rules.

Since medial is derivable in $\{\mathsf{c}\downarrow, \mathsf{w}\downarrow\}$, as shown in Thm. 3.13, the results we obtain for aSKSg hold also for $\mathsf{KSg} \cup \{\mathsf{c}\uparrow\}$, i.e. without medial. However since the derivation of medial contains depth 3 rule applications (the weakening steps), the result holds only for systems of depth greater than or equal to 3, which is somewhat less clean than the result for aSKSg.

Definition 4.4. Observe the following derivations in 1-aSKSg:

$$\mathsf{m}(0)\,\dfrac{\mathsf{c}\uparrow(1)\,\dfrac{A \vee (B \wedge C)}{(A \wedge A) \vee (B \wedge C)}}{[A \vee B] \wedge [A \vee C]} \qquad \mathsf{c}\downarrow(0)\,\dfrac{\mathsf{m}(0)\,\dfrac{(A \wedge B) \vee (A \wedge C)}{[A \vee A] \wedge [B \vee C]}}{A \wedge [B \vee C]}$$

$$\begin{array}{rc} & [A \vee B] \wedge [A \vee C] \\ =(0) & \overline{\qquad\qquad\qquad} \\ & [A \vee B] \wedge [C \vee A] \\ \mathsf{s}(1),\mathsf{s}(0) & \overline{\qquad\qquad\qquad} \\ & A \vee (B \wedge C) \vee A \\ =(1) & \overline{\qquad\qquad\qquad} \\ & A \vee A \vee (B \wedge C) \\ \mathsf{c}\downarrow(1) & \overline{\qquad\qquad\qquad} \\ & A \vee (B \wedge C) \end{array} \qquad \begin{array}{rc} & A \wedge [B \vee C] \\ \mathsf{c}\uparrow(0) & \overline{\qquad\qquad\qquad} \\ & A \wedge A \wedge [B \vee C] \\ =(0) & \overline{\qquad\qquad\qquad} \\ & A \wedge [B \vee C] \wedge A \\ 2\cdot\mathsf{s}(0) & \overline{\qquad\qquad\qquad} \\ & (A \wedge B) \vee (C \wedge A) \\ =(1) & \overline{\qquad\qquad\qquad} \\ & (A \wedge B) \vee (A \wedge C) \end{array}$$

From these we define four macro-rules, collectively known as the distributivity laws, which should be understood as abbreviations for the above derivations:

$$\mathsf{d}_2{\uparrow}\,\frac{A \vee (B \wedge C)}{[A \vee B] \wedge [A \vee C]} \qquad \mathsf{d}_2{\downarrow}\,\frac{(A \wedge B) \vee (A \wedge C)}{A \wedge [B \vee C]}$$

$$\mathsf{d}_1{\downarrow}\,\frac{[A \vee B] \wedge [A \vee C]}{A \vee (B \wedge C)} \qquad \mathsf{d}_1{\uparrow}\,\frac{A \wedge [B \vee C]}{(A \wedge B) \vee (A \wedge C)}$$

All instances of these macro-rules will be at depth 0, so when expanded will contain only at most depth 1 steps.

Like switch and medial, these rules can increase or decrease the depth of a formula. However, unlike switch and medial, all the above rules are invertible, indeed rules in the same column are inverse to each other, and rules in the same row are dual to each other. This invertibility allows us to "unfold" formulae at will, bringing subformulae out to whatever depth we wish, and then push them back down again. For example consider the following transformation from a derivation with depth 2 and 3 inference steps to a derivation with steps of depth at most 1:

$$\begin{array}{rc}
 & A \vee [A' \vee (B \wedge [C \vee C])] \\ \hline
\mathsf{c}{\downarrow}(2) & A \vee [A' \vee (B \wedge C)] \\ \hline
=(2) & A \vee [A' \vee (B \wedge [C \vee \bot])] \\ \hline
\mathsf{w}{\downarrow}(3) & A \vee [A' \vee (B \wedge [C \vee D])]
\end{array}
\quad \to \quad
\begin{array}{rc}
 & A \vee [A' \vee (B \wedge [C \vee C])] \\ \hline
=(0) & [A \vee A'] \vee (B \wedge [C \vee C]) \\ \hline
\mathsf{d}_2{\uparrow} & [[A \vee A'] \vee B] \wedge [[A \vee A'] \vee [C \vee C]] \\ \hline
\mathsf{c}{\downarrow}(1) & [[A \vee A'] \vee B] \wedge [[A \vee A'] \vee C] \\ \hline
=(1) & [[A \vee A'] \vee B] \wedge [[A \vee A'] \vee [C \vee \bot]] \\ \hline
\mathsf{w}{\downarrow}(1) & [[A \vee A'] \vee B] \wedge [[A \vee A'] \vee [C \vee D]] \\ \hline
\mathsf{d}_1{\downarrow} & [A \vee A'] \vee (B \wedge [C \vee D]) \\ \hline
=(0) & A \vee [A' \vee (B \wedge [C \vee D])]
\end{array}$$

Such a transformation preserves derivability, and so can be locally applied in a derivation to recursively reduce the depth of any inference step.

Theorem 4.5 (The depth-change trick)**.** *For every* aSKSg*-derivation* Φ *there is a* 1-aSKSg*-derivation* Ψ*, with the same premiss and conclusion, of size* $O(|\Phi|^3)$*.*

Proof. By commutativity we assume that every inference step operates in the right formula of a conjunction or disjunction. For each inference step ρ of depth at least 2 occurring in Φ, recursively apply the following transformations:

$$\rho(k)\,\frac{A \wedge (B \wedge C)}{A \wedge (B \wedge C')} \quad \to \quad \begin{array}{c} =(0)\,\dfrac{A \wedge (B \wedge C)}{(A \wedge B) \wedge C} \\ \rho(k)\,\dfrac{}{(A \wedge B) \wedge C'} \\ =(0)\,\dfrac{}{A \wedge (B \wedge C')} \end{array}$$

$$\rho(k)\,\frac{A \wedge [B \vee [C \vee D]]}{A \wedge [B \vee [C \vee D']]} \quad \to \quad \begin{array}{c} =(0)\,\dfrac{A \wedge [B \vee [C \vee D]]}{A \wedge [[B \vee C] \vee D]} \\ \rho(k)\,\dfrac{}{A \wedge [[B \vee C] \vee D']} \\ =(0)\,\dfrac{}{A \wedge [B \vee [C \vee D']]} \end{array}$$

$$\rho(k)\,\frac{A \wedge [B \vee (C \wedge D)]}{A \wedge [B \vee (C \wedge D')]} \quad \to \quad \begin{array}{c} \mathsf{d}_2\!\uparrow\,\dfrac{A \wedge [B \vee (C \wedge D)]}{A \wedge ([B \vee C] \wedge [B \vee D])} \\ =(0)\,\dfrac{}{(A \wedge [B \vee C]) \wedge [B \vee D]} \\ \rho(k-1)\,\dfrac{}{(A \wedge [B \vee C]) \wedge [B \vee D']} \\ =(0)\,\dfrac{}{A \wedge ([B \vee C] \wedge [B \vee D'])} \\ \mathsf{d}_1\!\downarrow\,\dfrac{}{A \wedge [B \vee (C \wedge D')]} \end{array}$$

Each transformation step strictly reduces the number of connectives in whose scope ρ occurs, and so this process terminates in at most n_ρ steps, where n_ρ is the size of ρ's conclusion, and ultimately yields a derivation Ψ_ρ containing only inference steps of depth at most 1.

At each step the width, i.e. maximum formula size, of the derivation expands by $O(n_\rho)$ and the length expands by a constant, so we have that $|\Psi_\rho|$ has width $O(n_\rho^2)$ and length $O(n_\rho)$, and so size $O(n_\rho^3)$. Finally, since $\sum_\rho n_\rho = |\Phi| - 1$, we have that $|\Psi| = O\left(\sum_\rho |\Psi_\rho|\right) = O\left(\sum_\rho n_\rho^3\right) = O(|\Phi^3|)$, as required □

Since medial steps can be replaced by 3-$\{\mathsf{w}\!\downarrow, \mathsf{c}\!\downarrow\}$-derivations and $\mathsf{aw}\!\uparrow$-steps can be eliminated from a KS^+-proof in linear time by Prop. 3.6, we obtain the equivalence of these three systems.

Corollary 4.6. 1-aSKSg, 3-$(\mathsf{KSg} \cup \{\mathsf{c}\!\uparrow\})$ *and* KS^+ *are polynomially equivalent.*

Also, by the quasipolynomial-time normalisation procedure of Chapt. 5, we have the following result:

Corollary 4.7. 1-aSKSg *and* 3-$(\mathsf{KSg} \cup \{\mathsf{c}\!\uparrow\})$ *quasipolynomially simulate Hilbert-Frege systems.*

4.2 Reduction to cut-free sequent calculus with elimination rules

Sequent calculi can essentially be considered depth 1 systems, since the relation between branches is conjunction and the comma is interpreted as disjunction. It is therefore possible to embed our systems into a cut-free sequent-like system, augmented slightly to give it some top-down symmetry. We present an example of such a system below based on the one-sided calculus called $\mathsf{GS1p}$ in [TS96].

Definition 4.8. A *sequent* is a multiset of formulae, denoted by the variables Γ, Δ, etc., with the comma ',' denoting multiset union. We omit braces $\{,\}$ when writing multisets.

The one-sided cut-free sequent calculus $\mathsf{GS1p}$ is defined by the rules below.

$$\top\text{-i}\,\frac{}{\top} \qquad \mathsf{id}\,\frac{}{A, \bar{A}} \qquad \mathsf{w}\,\frac{\Gamma}{\Gamma, A} \qquad \mathsf{c}\,\frac{\Gamma, A, A}{\Gamma, A} \qquad \vee\text{-i}\,\frac{\Gamma, A, B}{\Gamma, A \vee B} \qquad \wedge\text{-i}\,\frac{\Gamma, A \quad B, \Delta}{\Gamma, A \wedge B, \Delta}$$

$\top$-*intro*   *identity*   *weakening*   *contraction* $\vee$-*intro* $\wedge$-*intro*

We define the following elimination rules,

$$\wedge\text{-l}\,\frac{\Gamma, A \wedge B}{\Gamma, A} \qquad \wedge\text{-r}\,\frac{\Gamma, A \wedge B}{\Gamma, B} \qquad \vee\text{-e}\,\frac{\Gamma, A \vee B}{\Gamma, A, B} \qquad \bot\text{-e}\,\frac{\Gamma, \bot}{\Gamma}$$

left $\wedge$-*elim*   *right* $\wedge$-*elim* $\vee$-*elim* $\bot$-*elim*

and the system $\mathsf{GS1p}^+$ is obtained by augmenting $\mathsf{GS1p}$ with these rules.

A (dag-like) proof of a formula A in either of these systems is a list of sequents ending with A, where each sequent is the conclusion of an instance of an inference step of the system, all of whose premisses have occurred previously in the list. The axioms $\top$-i and id are considered as rules with 0 premisses.

Theorem 4.9. *(Dag-like)* $\mathsf{GS1p}^+$ *polynomially simulates* $\mathsf{1\text{-}aSKSg}$.

Proof. We translate proofs in the latter system by locally simulating each inference step in the former, construing surface $\vee$-symbols as commas. The result then follows by juxtaposing the simulations of each inference step in a CoS proof and then by appropriate applications of the $\vee$-i rule.

The translation of structural steps $\mathsf{i}\!\downarrow$, $\mathsf{w}\!\downarrow$ and $\mathsf{c}\!\downarrow$ and the rebracketing $=$ is routine, so we omit them. The two interesting cases are:

$$\mathsf{s}\,\frac{A \wedge [B \vee C]}{(A \wedge B) \vee C} \vee D \quad \rightarrow \quad \begin{array}{ll} A \wedge [B \vee C], D & \\ A, D & \wedge\text{-l} \\ B \vee C, D & \wedge\text{-r} \\ B, C, D & \vee\text{-e} \\ A \wedge B, C, D & \wedge\text{-i} \end{array}$$

$$\mathsf{c}\!\uparrow\,\frac{A}{A \wedge A} \vee B \quad \rightarrow \quad \begin{array}{ll} A, B & \\ A \wedge A, B, B & \wedge\text{-i} \\ A \wedge A, B & \mathsf{c} \end{array}$$

Finally, all the unit rules can be simulated in $\mathsf{GS1p}^+$ by contraction or weakening, except the following:

$$=\,\frac{A}{A \wedge \top} \vee B \quad \rightarrow \quad \begin{array}{ll} A, B & \\ \top & \top\text{-i} \\ A \wedge \top, B & \wedge\text{-i} \end{array}$$

$$=\,\frac{A \vee \bot}{A} \vee B \quad \rightarrow \quad \begin{array}{ll} A, \bot, B & \\ A, B & \bot\text{-e} \end{array}$$

□

Corollary 4.10. $\mathsf{GS1p}^+$ *quasipolynomially simulates Hilbert-Frege systems.*

4.3 Some remarks

It is a natural question to ask whether the results of this chapter can be transferred to KSg, i.e. in the absence of cocontraction. At first glance this seems unlikely since derivability of the distributivity rules $\mathsf{d}_1\!\uparrow$ and $\mathsf{d}_2\!\uparrow$ rely on the presence of $\mathsf{c}\!\uparrow$, but there is no fundamental reason why the same result could not be achieved by other means, for example by some global transformation of proofs. Proving the impossibility of this, i.e. finding superpolynomial lower bounds for bounded KSg systems, might serve as a starting point towards finding nontrivial lower bounds for KS.

Cut-free sequent calculi are suitable systems in which to conduct proof search, due to their analyticity: any formula occurring in a proof is a subformula of its conclusion. However the complexity of proof search in a system is bounded below by the size of proofs in that system, and so proof search in cut-free sequent systems is necessarily inefficient on tautologies such as the pigeonhole principle, which we present in Chapt. 9, that have only exponential-size cut-free proofs. One could argue that the system $\mathsf{GS1p}^+$ defined above might be a better base for proof search for such tautologies, due to its quasipolynomial simulation of Hilbert-Frege proofs, while still retaining some of the structural advantages of cut-free sequent systems.

At the same time, due to the relative simplicity of $\mathsf{GS1p}^+$ and controlled way in which it breaks the subformula property, it is perhaps pertinent to pursue nontrivial lower bounds for this system, which could then be transferred to Hilbert-Frege systems by the quasipolynomial simulation result.

Chapter 5

Quasipolynomial-time normalisation to KS^+

One of the surprising properties of deep inference systems, arguably, is that 'cut-elimination' can be done in quasipolynomial time, rather than the necessarily exponential time taken by Gentzen cut-elimination. By translation to the sequent calculus this corresponds to the fact that the part of cut-elimination that eliminates cuts between descendants of negation-steps can be carried out in quasipolynomial-time. This fragment is equivalent to the monotone sequent calculus and, indeed, these results were first proved by Atserias et al. for that system [AGP02], before Jeřábek observed that they could be translated to deep inference [Jeř09].

The material in this chapter is based on the work of Atserias et al. [AGP02], Jeřábek [Jeř09] and Bruscoli et al. [BGGP10]. In fact, our construction is essentially a simplified version of that appearing in [BGGP10].

For this chapter we will use some shorthand notations for substitutions. If A and B are formulae then we write $A[B/a]$ to denote the formula obtained by substituting every occurrence of a in A by B. We define $\Phi[B/a]$ and $A[\Phi/a]$ similarly for a derivation Φ. Note that this substitution is *positive*: it affects only occurrences of a and not $\bar{a}$, so that occurrences of $\bar{a}$ remain intact and are not replaced by $\bar{B}$.

We write $A[B_i/a_i]_{i=1}^n$ as shorthand for $A[B_1/a_1][B_2/a_2]\cdots[B_n/a_n]$, and write $\boldsymbol{a}$ or $(a_i)_{i=1}^n$ to denote a vector of atoms $(a_1,\dots,a_n)$.

The aim of this chapter is to prove the following theorem:

Theorem 5.1. *For every SKS-proof Φ with conclusion τ over n propositional variables, there is an KS^+-proof of τ of size $|\Phi|^{O(1)} \cdot n^{O(\log n)}$.*

5.1 Monotone formulae computing threshold functions

The *threshold functions* TH^n_k are boolean functions $\{0,1\}^n \to \{0,1\}$ with $\mathsf{TH}^n_k(\sigma) = 1$ just if $\sum\limits_{i=1}^{n} \sigma_i \geq k$. Clearly these functions are monotone, and so we can construct formulae over $\{\top, \bot, \wedge, \vee\}$ that compute them, by adequacy. Using a divide-and-conquer strategy it is not difficult to obtain such formulae of size quasipolynomial in n, for example in the definition below that is standard in the literature.

Definition 5.2. We define the formulae $\mathsf{th}^n_k(a_1, \dots, a_n)$, for n a power of 2, as follows:

$$\mathsf{th}^1_k(a) \quad := \begin{cases} \top & k = 0 \\ a & k = 1 \\ \bot & k > 1 \end{cases}$$

$$\mathsf{th}^{2n}_k(\boldsymbol{a}, \boldsymbol{b}) \quad := \bigvee_{i+j=k} \mathsf{th}^n_i(\boldsymbol{a}) \wedge \mathsf{th}^n_j(\boldsymbol{b})$$

Remark 5.3. Note that we have only defined th^n_k when n is a power of 2. We will always assume this is the case, and consider proofs over atoms $a_1, \dots, a_m$ to also trivially be over $n - m$ further atoms, for n the least power of 2 greater than or equal to m. This only blows up our constructions by at most multiplication with $n^{O(1)}$.

th^n_k clearly computes TH^n_k and it is not difficult to see that th^n_k has size $n^{\log n - \log\log n + O(1)}$ using analytic methods.[1] It is this bound that essentially determines the complexity of the $\mathsf{ai}{\uparrow}$-elimination procedure later in this chapter.

These threshold formulae have been used in previous works, e.g. [AGP02] [Jeř12] [BGGP10], to obtain similar complexity results, but there are several other constructions one could consider. In fact it is known that polynomial-size monotone threshold formulae exist, due to the $O(\log n)$-depth AKS sorting networks [AKS83] and also an elegant probabilistic construction by Valiant [Val84], however it is not known whether formal proofs corresponding to basic properties of threshold functions can be constructed efficiently.[2]

[1]Induct over n with the hypothesis $|\mathsf{th}^n_k| \leq \frac{n \cdot k^{\log n}}{(\log n)!}$, using the upper integral bound for the identity $|\mathsf{th}^n_k| = 2 \cdot \sum\limits_{i=0}^{k} |\mathsf{th}^{n/2}_i|$, then substitute $k = \frac{n}{2}$ and simplify. This bound is tight for $|\mathsf{th}^n_{n/2}|$, by using the lower integral bound.

[2]We should point out that Jeřábek has made significant progress in this direction by formalising the AKS construction in a theory of bounded arithmetic, conditional on the provability of existence of suitable expander graphs in this theory [Jeř11]. By proving basic properties of these networks in the theory one can construct propositional proofs of the necessary sequents in the dag-like monotone sequent calculus.

In any case, even within the divide-and-conquer paradigm, one can design threshold formulae differently, for example using the following identity,

$$\mathsf{TH}^n_{2k}(a_1, \ldots, a_n) = \bigvee_{l+m=n} \mathsf{TH}^l_k(a_1, \ldots, a_l) \wedge \mathsf{TH}^m_k(a_{l+1}, \ldots, a_n)$$

which conducts a divide-and-conquer on the threshold rather than the number of inputs, encoding the statement "some point along $\boldsymbol{a}$ has half of the 1s on its left and half on its right". Formulae obtained from this identity have the same size as th^n_k, up to addition of a constant in the exponent.

One could conceive of more sophisticated constructions that divide-and-conquer on both parameters:

$$\mathsf{TH}^{2n}_{2k}(a_1, \ldots, a_{2n}) = \bigvee_{i=1}^{n} \mathsf{TH}^n_k(a_i, \ldots, a_{i+n-1}) \wedge \mathsf{TH}^n_k(a_1, \ldots, a_{i-1}, a_{i+n}, \ldots, a_n)$$

It is helpful here to think of the variables in a circle, with the identity encoding the statement "some semicircle and its complement each contain half of the 1s", which is necessarily true by the intermediate value theorem. The formulae resulting from this construction have size $\leq n^{\frac{1}{2}\log n - \frac{1}{2}}$,[3] and so exhibit a near-quadratic improvement in efficiency over the previous two constructions,[4] although proofs we have so far constructed of basic threshold properties for this construction are somewhat more involved, to the extent that using these for normalisation does not deliver any overall improvement in complexity.

5.2 Basic properties of the threshold functions

In this section we construct formal proofs witnessing some basic properties of the threshold functions, for the encoding th^n_k, which we rely on in the next section.

Definition 5.4. An $\mathsf{SKS} \setminus \{\mathsf{ai}\!\downarrow, \mathsf{ai}\!\uparrow\}$-derivation is called *monotone*.

Most of the proofs in this chapter are inductions, and for the base cases it will suffice to build any monotone proof of a single formula or simple class of formulae. For this reason we will implicitly assume the following result when omitting base cases of inductions.

[3]Let $S(n,k)$ denote the size of the formula obtained with n variables and threshold k. Notice that $S(n,k) = n \cdot S(\frac{n}{2}, \frac{k}{2})$ and simply evaluate this identity. The bound is an exact equality for $S(n, \frac{n}{2})$.

[4]Incidentally, a basic counting argument shows that this is an optimal construction, up to addition of a constant in the exponent, for a large class of divide-and-conquer definitions.

Proposition 5.5 (Monotone implicational completeness)**.** *Let A, B be negation-free formulae such that $A \to B$ is valid. Then there is a monotone derivation $\begin{array}{c} A \\ \| \\ B \end{array}$.*

Proof sketch. Construct a disjunctive normal form A' of A and conjunctive normal form B' of B, using the distributivity laws in Dfn. 4.4 so that there are monotone derivations $\begin{array}{c} B' \\ \| \\ B \end{array}$ and $\begin{array}{c} A \\ \| \\ A' \end{array}$. Clearly each conjunction of A' logically implies each disjunction of B' and so there must be a nonempty intersection of their variables, by monotonicity, whence we can construct derivations in $\{\mathsf{aw}\downarrow, \mathsf{aw}\uparrow\}$ witnessing this fact. Using these derivations and applying $\mathsf{c}\downarrow$ and $\mathsf{c}\uparrow$ appropriately we can construct a monotone derivation $\begin{array}{c} A' \\ \| \\ B' \end{array}$, whence the result follows by sequential composition of these derivations. □

Proposition 5.6. *There are monotone derivations of size $n^{O(\log n)}$ of the form,*

1. $\begin{array}{c} \top \\ \| \\ \mathsf{th}^n_0(\boldsymbol{a}) \end{array}$

2. $\begin{array}{c} \mathsf{th}^n_k(\boldsymbol{a}) \\ \| \\ \bot \end{array}$*, where $k > n$.*

3. $\begin{array}{c} \mathsf{th}^n_k(\boldsymbol{a})[\bot/a_i] \\ \| \\ \mathsf{th}^n_{k+1}(\boldsymbol{a})[\top/a_i] \end{array}$

where $\boldsymbol{a}$ is a vector of atoms of length n.

Proof. Notice that both directions of the equations $A \vee \top = \top$ and $A \wedge \bot = \bot$ can be derived in $\{\mathsf{aw}\downarrow, \mathsf{aw}\uparrow\}$. By Prop. 7.4 we have that the rewriting system obtained by augmenting the $=$-rules with these equations reduces every formula in linear time to a unit-free formula or $\top$ or $\bot$. The resulting equational theory preserves the boolean function computed by a formula, and so we can construct appropriate derivations for 1 and 2 in $\{\mathsf{aw}\downarrow, \mathsf{aw}\uparrow\}$, since th^n_0 is constantly true and th^n_k is constantly false for $k > n$.

For 3, we construct appropriate derivations by induction on the number of atoms n. Without loss of generality we assume that a_i is in the first half of the variables and construct the required derivation as follows,

$$= \frac{\mathsf{th}^{2n}_k(\boldsymbol{a},\boldsymbol{b})[\bot/a_i]}{\dfrac{\bigvee_{i+j=k} \left(\begin{array}{c} \mathsf{th}^n_i(\boldsymbol{a})[\bot/a_i] \\ IH \Big\Vert \\ \mathsf{th}^n_{i+1}(\boldsymbol{a})[\top/a_i] \end{array} \wedge \mathsf{th}^n_j(\boldsymbol{b}) \right) \vee \mathsf{w}{\downarrow} \dfrac{\bot}{\mathsf{th}^n_0(\boldsymbol{a})[\top/a_i] \wedge \mathsf{th}^n_{k+1}(\boldsymbol{b})}}{\mathsf{th}^{2n}_{k+1}(\boldsymbol{a},\boldsymbol{b})[\top/a_i]} =}$$

where the derivation marked IH is obtained by the inductive hypothesis. □

5.3 Cut-elimination procedure

We now describe a quasipolynomial-time procedure for eliminating $\mathsf{ai}\uparrow$-steps in an SKS-proof of a tautology τ. Working in KS^+, the basic idea is to begin by not assuming that any of the variables are true and then, on the assumption that τ is not true, use progressive runs of a slightly modified monotone version of the input proof to deduce that more and more variables are true. Eventually we have that there are more truths than variables, which is absurd, whence the truth of τ follows.

Proposition 5.7. *For every SKS-proof Φ with conclusion τ over atoms $a_1, \ldots, a_n$, there is a monotone derivation of the following form,*

$$\begin{array}{c} \bigwedge_{i=1}^{n} [a_i \vee \bar{a}_i] \\ \Phi' \Big\Vert \\ \tau \vee \bigvee_{i=1}^{n} (a_i \wedge \bar{a}_i) \end{array}$$

of size $|\Phi|^{O(1)}$.

Proof. Let Φ be in CoS form and proceed by induction on its length.

In the base case we have $\Phi \equiv \tau \equiv \top$, so A monotonically implies $\tau \vee B$ for any formulae A, B, whence the result follows by monotone implicational completeness.

When Φ is obtained by extending some proof Ψ by an inference step ρ, where

$\rho \notin \{\mathsf{ai}\!\downarrow, \mathsf{ai}\!\uparrow\}$, then we conduct the following transformation,

$$\begin{array}{c} \Psi \Big\| \mathsf{SKS} \\ \rho\,\dfrac{\xi\{A\}}{\xi\{B\}} \end{array} \quad \to \quad \begin{array}{c} \bigwedge\limits_{i=1}^{n} [a_i \vee \bar{a}_i] \\ \Psi' \Big\| \\ \xi\left\{ \rho\,\dfrac{A}{B} \right\} \vee \bigvee\limits_{i=1}^{n} (a_i \wedge \bar{a}_i) \end{array}$$

where Ψ' is obtained from Ψ by the inductive hypothesis.

Otherwise Φ is obtained by extending some proof Ψ by either an $\mathsf{ai}\!\downarrow$-step or a $\mathsf{ai}\!\uparrow$-step. In the former case we transform Φ as follows,

$$\begin{array}{c} \Psi \Big\| \mathsf{SKS} \\ \mathsf{ai}\!\downarrow\dfrac{\xi\{\top\}}{\xi\{a_k \vee \bar{a}_k\}} \end{array} \quad \to \quad =\dfrac{\bigwedge\limits_{i=1}^{n} [a_i \vee \bar{a}_i]}{=\dfrac{\mathsf{c}\!\uparrow\dfrac{a_k \vee \bar{a}_k}{[a_k \vee \bar{a}_k] \wedge [a_k \vee \bar{a}_k]} \wedge \bigwedge\limits_{i \neq k} [a_i \vee \bar{a}_i]}{\mathsf{s}\,\dfrac{[a_k \vee \bar{a}_k] \wedge \begin{array}{c} \bigwedge\limits_{i=1}^{n} [a_i \vee \bar{a}_i] \\ \Psi' \Big\| \\ \xi\{\top\} \vee \bigvee\limits_{i=1}^{n} (a_i \wedge \bar{a}_i) \end{array}}{\begin{array}{c} [a_k \vee \bar{a}_k] \wedge \xi\{\top\} \\ \bullet \Big\| \{\mathsf{s}\} \\ \xi\{a_k \vee \bar{a}_k\} \end{array} \vee \bigvee\limits_{i=1}^{n} (a_i \wedge \bar{a}_i)}}}$$

where Ψ' is obtained by the inductive hypothesis and $\bullet$ by Lemma 3.11. In the latter case we transform Φ as follows,

$$\begin{array}{c} \Psi \Big\| \mathsf{SKS} \\ \mathsf{ai}\!\uparrow\dfrac{\xi\{a_k \wedge \bar{a}_k\}}{\xi\{\bot\}} \end{array} \quad \to \quad \begin{array}{c} \bigwedge\limits_{i=1}^{n} [a_i \vee \bar{a}_i] \\ \Psi' \Big\| \\ =\dfrac{\begin{array}{c} \xi\{a_k \wedge \bar{a}_k\} \\ \bullet \Big\| \{\mathsf{s}\} \\ \xi\{\bot\} \vee (a_k \wedge \bar{a}_k) \end{array} \vee =\dfrac{\bigvee\limits_{i=1}^{n} (a_i \wedge \bar{a}_i)}{(a_k \wedge \bar{a}_k) \vee \bigvee\limits_{i \neq k} (a_i \wedge \bar{a}_i)}}{\xi\{\bot\} \vee =\dfrac{\mathsf{c}\!\downarrow\dfrac{(a_k \wedge \bar{a}_k) \vee (a_k \wedge \bar{a}_k)}{a_k \wedge \bar{a}_k} \vee \bigvee\limits_{i \neq k} (a_i \wedge \bar{a}_i)}{\bigvee\limits_{i=1}^{n} (a_i \wedge \bar{a}_i)}} \end{array}$$

where Ψ' is obtained by the inductive hypothesis and $\bullet$ by Lemma 3.11. $\square$

Proposition 5.8. *There are derivations* $\begin{array}{c}\mathsf{th}^n_k(\boldsymbol{a})\\ \Vert\{\mathsf{ai}\downarrow,\mathsf{ac}\downarrow,\mathsf{s}\}\\ a_i \vee (\bar{a}_i \wedge \mathsf{th}^n_k(\boldsymbol{a})[\bot/a_i])\end{array}$ *of size* $n^{O(\log n)}$.

Proof. We give the derivations below,

$$
=\frac{\begin{array}{c}\mathsf{th}^n_k(\boldsymbol{a})\\ \bullet\Vert\{\mathsf{s}\}\\ a_i \vee \mathsf{th}^n_k(\boldsymbol{a})[\bot/a_i]\end{array}}{=\dfrac{a_i \vee \left(= \dfrac{\mathsf{th}^n_k(\boldsymbol{a})[\bot/a_i]}{\mathsf{s}\dfrac{\mathsf{ai}\downarrow\dfrac{\top}{a_i \vee \bar{a}_i} \wedge \mathsf{th}^n_k(\boldsymbol{a})[\bot/a_i]}{a_i \vee (\bar{a}_i \wedge \mathsf{th}^n_k(\boldsymbol{a})[\bot/a_i])}}\right)}{\mathsf{ac}\downarrow\dfrac{a_i \vee a_i}{a_i} \vee (\bar{a}_i \wedge \mathsf{th}^n_k(\boldsymbol{a})[\bot/a_i])}}
$$

where the derivation marked $\bullet$ is obtained by Lemma 3.11. $\square$

At this point we can replace $[a_i \vee \bar{a}_i]$, for each i, in the premiss of the derivation Φ' obtained by Prop. 5.7 by derivations of $a_i \vee (\bar{a}_i \wedge \mathsf{th}^n_k(\boldsymbol{a})[\bot/a_i])$ from $\mathsf{th}^n_k(\boldsymbol{a})$ given in Prop. 5.8 above. Propagating this substitution for $\bar{a}_i$ through Φ' allows us to increase the working threshold to $k+1$. We state this formally in the result below.

Lemma 5.9. *If* Φ *is an* SKS*-proof of a tautology* τ *over atoms* $\boldsymbol{a} = (a_1, \ldots, a_n)$ *then there is a derivation* $\begin{array}{c}\mathsf{th}^n_k(\boldsymbol{a})\\ \Vert\mathsf{KS}^+\\ \tau \vee \mathsf{th}^n_{k+1}(\boldsymbol{a})\end{array}$ *of size* $|\Phi|^{O(1)} \cdot n^{O(\log n)}$.

Proof. Let Φ' be the monotone derivations from $\bigwedge_i^n [a_i \vee \bar{a}_i]$ to $\tau \vee \bigvee_i^n (a_i \wedge \bar{a}_i)$ obtained by

Prop. 5.7. We give an explicit construction of the required derivations below,

$$
\begin{array}{c}
\mathsf{c}\!\uparrow \dfrac{\mathsf{th}^n_k(\boldsymbol{a})}{\left(\begin{array}{c}\mathsf{th}^n_k(\boldsymbol{a})\\ \bullet\Vert \\ a_1 \vee (\bar{a}_1 \wedge \mathsf{th}^n_k\,(\boldsymbol{a})\,[\bot/a_1])\end{array} \wedge \cdots \wedge \begin{array}{c}\mathsf{th}^n_k(\boldsymbol{a})\\ \bullet\Vert \\ a_n \vee (\bar{a}_n \wedge \mathsf{th}^n_k\,(\boldsymbol{a})\,[\bot/a_n])\end{array}\right)} \\
\Big\Vert \, \Phi'\big[(\bar{a}_i \wedge \mathsf{th}^n_k(\boldsymbol{a})[\bot/a_i])/\bar{a}_i\big]^n_{i=1} \\
\mathcal{T}\left[\left(= \dfrac{\bar{a}_i \wedge \mathsf{w}\!\uparrow \dfrac{\mathsf{th}^n_k(\boldsymbol{a})[\bot/a_i]}{\top}}{\bar{a}_i}\right)\Big/ \bar{a}_i\right]^n_{i=1} \vee \mathsf{c}\!\downarrow \dfrac{\bigvee_{i=1}^{n}\left(= \dfrac{a_i \wedge \mathsf{aw}\!\uparrow \dfrac{\bar{a}_i}{\top}}{a_i} \wedge \begin{array}{c}\mathsf{th}^n_k(\boldsymbol{a})[\bot/a_i]\\ \circ\Vert \\ \mathsf{th}^n_{k+1}(\boldsymbol{a})[\top/a_i]\end{array}\right)}{\mathsf{th}^n_{k+1}(\boldsymbol{a})}
\end{array}
$$

where the derivations marked $\bullet$ are obtained by Prop. 5.8 and the derivation marked $\circ$ is obtained by Prop. 5.6. □

Now we can simply stitch together $n+1$ of these derivations, one for each threshold value, to obtain a proof of the main theorem of this section.

Proof of Thm. 5.1. Inductively applying Lemma 5.9 for increasing values of the threshold k, we construct appropriate KS^+-derivations as follows,

$$
\begin{array}{c}
= \dfrac{\begin{array}{c}\bullet\Vert \\ \mathsf{th}^n_0(\boldsymbol{a}) \\ \Vert \\ \left[\mathcal{T} \vee \begin{array}{c}\mathsf{th}^n_1(\boldsymbol{a}) \\ \Vert \\ \left[\cdots \begin{array}{c}\ddots \\ \Vert \\ \left[\mathcal{T} \vee \begin{array}{c}\mathsf{th}^n_n(\boldsymbol{a}) \\ \Vert \\ \left[\mathcal{T} \vee \begin{array}{c}\mathsf{th}^n_{n+1}(\boldsymbol{a}) \\ \circ\Vert \\ \bot\end{array}\right]\end{array}\right]\end{array} \cdots\right]\end{array}\right]\end{array}}{\mathcal{T} \vee \cdots \vee \mathcal{T}} \\
\mathsf{c}\!\downarrow \dfrac{}{\mathcal{T}}
\end{array}
$$

where the derivations marked $\bullet$ and $\circ$ are obtained by Prop. 5.6. □

Corollary 5.10. $\mathsf{KS} \cup \{\mathsf{ac}\!\uparrow\}$ *quasipolynomially simulates Hilbert-Frege systems.*

Proof. Immediate from Props. 3.6, 3.16 and Thm. 5.1. □

5.4 Some remarks

Is normalisation from SKS to KS^+ possible in polynomial time? A positive answer might be obtained by analogous results for the monotone sequent calculus. We note that some progress has been made by Jeřábek in [Jeř11] by formalising the AKS sorting networks in a theory of bounded arithmetic, but the monotone proofs of basic threshold properties extracted from this formalisation are crucially dag-like, and so do not immediately translate into small KS^+-proofs in deep inference.

Another approach might be to show *existence* of small proofs, using probabilistic methods as in Valiant's work [Val84], although constructing an appropriate probability space in which to conduct this argument seems to require some nontrivial insight.

Part III

Analysing the logical and structural fragments

Chapter 6

Structural manipulations of proofs in deep inference

In this chapter we restrict our attention to the structural rules of SKS and consider reductions of situations when two structural steps overlap.

Similar situations arise in the sequent calculus, e.g. the derivation on the left below, where the conclusion of a weakening step overlaps with the premiss of a contraction step, can be reduced to the derivation on the right:

$$\begin{array}{r@{\;}c} & \Gamma, A \\ w & \overline{\Gamma, A, A} \\ c & \overline{\Gamma, A} \end{array} \qquad \to \qquad \Gamma, A$$

The intuition here is that if a formula introduced by weakening is later contracted then it is somewhat redundant. Unfortunately, in the sequent calculus, this intuition cannot always be realised, e.g, in the derivation below:

$$\begin{array}{r@{\;}c} & [A \vee B] \wedge C, A \\ w & \overline{[A \vee B] \wedge C, A, B} \\ \vee & \overline{[A \vee B] \wedge C, A \vee B \quad C} \\ \wedge & \overline{[A \vee B] \wedge C, [A \vee B] \wedge C} \\ c & \overline{[A \vee B] \wedge C} \end{array}$$

One occurrence of the formula B originates from a weakening step, and has a descendant that is later contracted, but there is no way to manipulate this derivation to get rid of the extra occurrence of B, since the occurrence of B it is paired with already occurs inside a larger formula at the top of the derivation.

The blockade is, of course, caused by the logical steps in the middle of the derivation; for the sequent calculus our intuition only applies when the formula being contracted is a *direct* descendant of a formula originating from a weakening step.

In deep inference, on the other hand, such redundancy can always be eliminated, e.g. this is how the above situation could be dealt with:

$$\mathsf{c}{\downarrow}\dfrac{\mathsf{s}\dfrac{\left[([A\vee B]\wedge C)\vee \mathsf{w}{\downarrow}\dfrac{A}{A\vee B}\right]\wedge C}{([A\vee B]\wedge C)\vee([A\vee B]\wedge C)}}{[A\vee B]\wedge C} \qquad\to\qquad \mathsf{m}\dfrac{\mathsf{s}\dfrac{[(([A\vee B]\wedge C)\vee A]\wedge C}{([A\vee B]\wedge C)\vee(A\wedge C)}}{\left[\mathsf{c}{\downarrow}\dfrac{A\vee A}{A}\vee B\right]\wedge \mathsf{c}{\downarrow}\dfrac{C\vee C}{C}}$$

Such manipulations are possible because the structural rules of deep inference generally commute with the logical rules. Due to atomicity we need not distinguish between direct and indirect descendants, and these sorts of reduction are dependent only on the structural behaviour of an atom, not how it interacts with logical steps.

To highlight this point we introduce *atomic flows*, a geometric abstraction of derivations that records precisely the structural information and ignores logical information, in order to reason about the interactions between various structural steps.

We define graph rewriting rules on atomic flows that correspond to sound manipulations of derivations and analyse the complexity such transformations. In particular we focus on a system of local rewriting rules that allows us to transform KS^+-proofs to KS-proofs of the same conclusion, with complexity determined by the number of certain types of paths in the initial flow. We will appeal to these results throughout Part IV, where we address the proof complexity of KS.

We point out that the results of this chapter are independent of the choice of logical rules; they hold for any system with the same structural rules and any set of logical rules that can derive switch and medial.

The material in this chapter is based on work that has appeared in [Das12] and also work currently in submission.

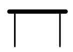
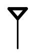
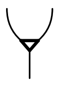
6.1 Atomic flows

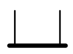
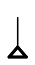
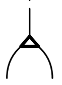
Gundersen and Guglielmi first developed atomic flows for SKS-derivations in [GG08], primarily for the purpose of designing normalisation procedures that were entirely controlled by consideration of the structural behaviour of a proof, not appealing to any logical information. In that work atomic flows were defined as combinatorial objects, essentially labelled directed graphs with different types of nodes, although we give a less formal definition, to avoid clutter.

Unfortunately we were unable to identify a textbook on graph rewriting to which a reader could be directed for background. However it is not difficult to see that many of the usual results of term rewriting, for example as found in [Klo92], apply in these settings too. Nonetheless we give informal definitions and arguments when we use such results.

We refer the reader to the initial paper [GG08], and also Gundersen's thesis [Gun09], for a comprehensive account of atomic flows and related constructions.

Definition 6.1 (Atomic flows). For an SKS-derivation Φ we define its *atomic flow*, $fl(\Phi)$, to be the diagram obtained by tracing the path of each atom through the derivation Φ, designating the creation, duplication and destruction of atoms by the following corresponding nodes:

$\mathsf{ai}{\downarrow}\,\dfrac{\top}{a \vee \bar{a}} \;\to$  $\mathsf{aw}{\downarrow}\,\dfrac{\bot}{a} \;\to$  $\mathsf{ac}{\downarrow}\,\dfrac{a \vee a}{a} \;\to$ 

$\mathsf{ai}{\uparrow}\,\dfrac{a \wedge \bar{a}}{\bot} \;\to$  $\mathsf{aw}{\uparrow}\,\dfrac{a}{\top} \;\to$  $\mathsf{ac}{\uparrow}\,\dfrac{a}{a \wedge a} \;\to$ 

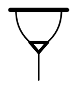
We do not have nodes for s, m or $=$ since they do not create, destroy or duplicate any atom occurrences.

More generally an atomic flow, not necessarily of a derivation, is a (vertically) directed graph embedded in the plane generated from the six types of node above.

Atomic flows are considered equivalent up to continuous deformation preserving the (vertical) ordering of connected edges. Note that edges may be *pending* at either end.

We define the size of a flow ϕ, denoted $|\phi|$, to be its number of edges.

In previous works atomic flows have been equipped with a labelling of the edges, or a polarity assignment, for example to avoid the following impossible situation:



Since we are only concerned with the complexity of flows and their transformations we do not include this extra structure; this does not affect the soundness or termination of our rewriting systems, and in fact is crucial in order to obtain confluence, for which labellings of edges can cause problems.

We do, however, insist that edges are vertically directed and so we are often able to talk about one node being 'above' another node. Notice that this order is not gen-

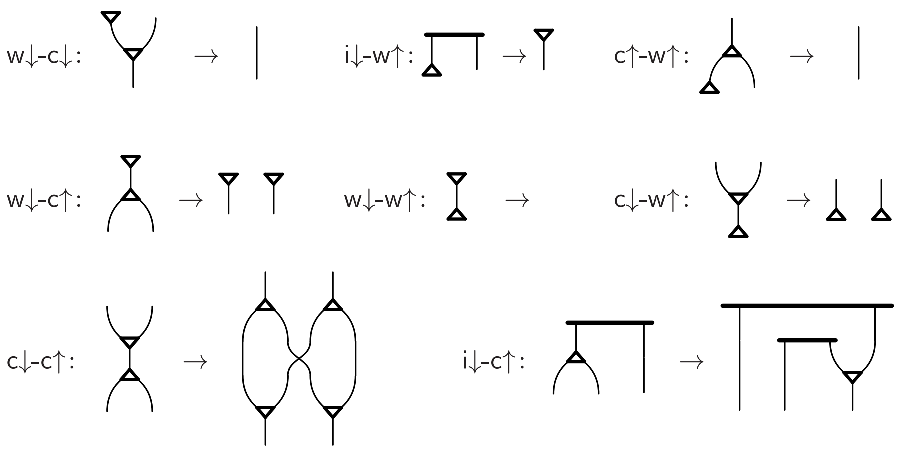


Figure 6-1: Local rewriting rules for the system norm.

erally preserved under deformation, e.g. if two nodes are in disconnected components. Whenever we use this notion in arguments it should be clear that it is being used correctly.

Definition 6.2. A *flow rewriting rule* is an ordered pair of flows, written $\phi \to \psi$. A *flow rewriting system* (FRS) is a set of flow rewriting rules. A *one-step reduction* of a flow ϕ in an FRS r is a flow ψ that is obtainable from ϕ by replacing some induced subgraph that is the left hand side of some rule in r with its right hand side.

Definition 6.3. We define a rewriting system norm on atomic flows in Fig. 6-1.

The system norm is essentially the system $\mathsf{c} \cup \mathsf{w}$ in [GG08], without the rules for i↑. The proof of termination that follows is similar to that for cycle-free flows in [GG08], and the proof of confluence is similar to that in [GGS10].

Notation 6.4. Let r be an FRS. We use the following notation:

1. We write $\phi \underset{\mathsf{r}}{\to} \psi$ if there is a one-step reduction from a flow ϕ to a flow ψ in r.
2. We denote by $\overset{*}{\underset{\mathsf{r}}{\to}}$ the reflexive transitive closure of $\underset{\mathsf{r}}{\to}$.
3. If a flow ϕ has a unique normal form under r we denote it by $\phi{\downarrow_{\mathsf{r}}}$.

In all cases we might omit the subscript r if it is clear from context.

Example 6.5. We give an example of a flow associated with a derivation in Fig. 6-2, as well as a reduction under norm, as defined in Dfn. 6.3, applying w↑ rules first.

The first equality follows by the definition of a flow, the second by deformation and the final by definition again. The intermediate steps are as follows:

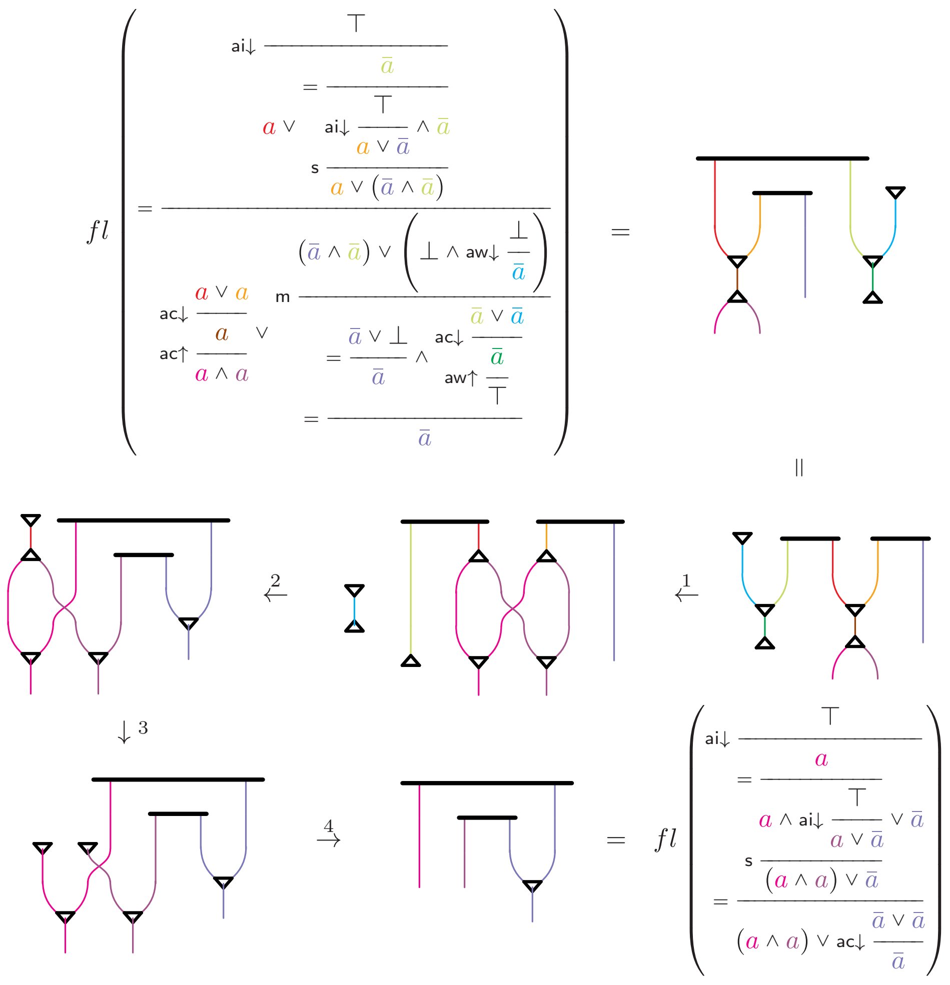


Figure 6-2: A proof, its flow and a reduction under norm.

1. Apply c↓-w↑ on the left and c↓-c↑ on the right.

2. Apply w↓-w↑ on the left, i↓-w↑ in the middle and i↓-c↑ on the right.

3. Apply w↓-c↑ on the left.

4. Apply w↓-c↓ twice on the left.

The use of colours in the initial and final flow identifies which edges corresponds to which atoms; in the intermediate flows the colours should aid the reader in reconstructing the corresponding transformations on the derivation.

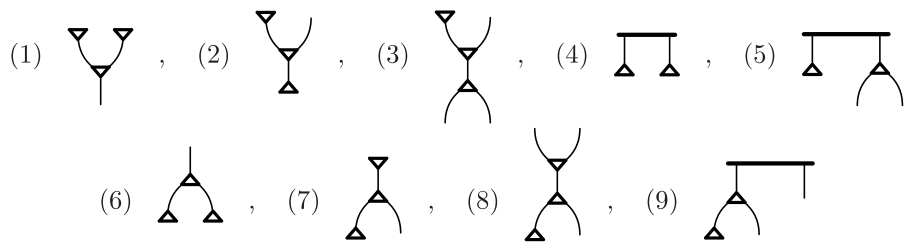
We now proceed to prove that reducing under norm is terminating and confluent. For this we use the usual critical pair lemma, that one only needs to check that the one-step contracta of every overlapping pair of redexes are joinable in order to conclude local confluence. It is simple to see that this is still valid in the flow rewriting setting, since the only other case to check for flows is when the redexes are disjoint, which is trivial. We point out that Newman's lemma, that any locally confluent terminating system is confluent, is true more generally for any binary relation on any set, and so indeed holds in this setting.

Theorem 6.6. $\underset{\mathsf{norm}}{\longrightarrow}$ *is terminating and confluent.*

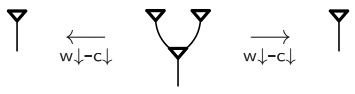
Proof. For a node ν in a flow ϕ, let $d(\nu, \phi)$ be the distance of ν from the top of ϕ, i.e. the minimum length of a (vertically) directed path from ν to an $\mathsf{ai}{\downarrow}$-node, an $\mathsf{aw}{\downarrow}$-node or an edge with upper end pending. For an atomic structural rule ρ, let $D(\rho, \phi)$ be the sequence of natural numbers that counts how many ρ-nodes in ϕ have each d-value, i.e. the sequence (n_i) such that, for each i, n_i is the number of ρ-nodes ν in ϕ with $d(\nu, \phi) = i$, and consider the lexicographical ordering $<$ on such sequences, with $(m_i) < (n_i)$ just if $m_k < n_k$ for some k and $m_i \leq n_i$ for $i > k$.

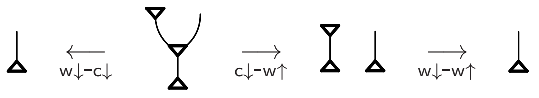
Clearly the rules $\mathsf{c}{\downarrow}$-$\mathsf{c}{\uparrow}$, $\mathsf{i}{\downarrow}$-$\mathsf{c}{\uparrow}$ and $\mathsf{w}{\downarrow}$-$\mathsf{c}{\uparrow}$ strictly reduce the $D(\mathsf{ac}{\uparrow}, \cdot)$-value of a flow, while the other rules of norm (as well as $\mathsf{w}{\downarrow}$-$\mathsf{c}{\uparrow}$) strictly reduce a flow's size, while not increasing the $D(\mathsf{ac}{\uparrow}, \cdot)$-value. Therefore each application of a norm rule strictly reduces the lexicographical product $D(\mathsf{ac}{\uparrow}, \cdot) \times |\cdot|$.

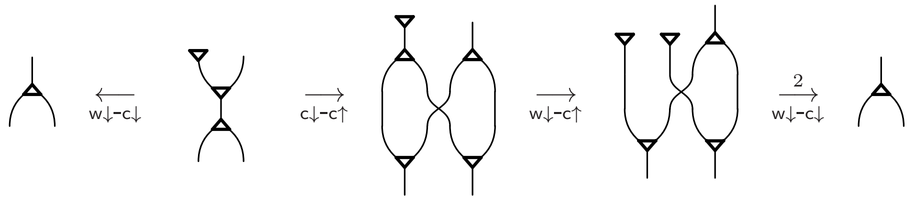
Since norm is terminating, it suffices to check local confluence for the critical pairs, which are the following by inspection:

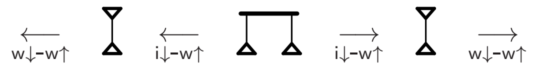


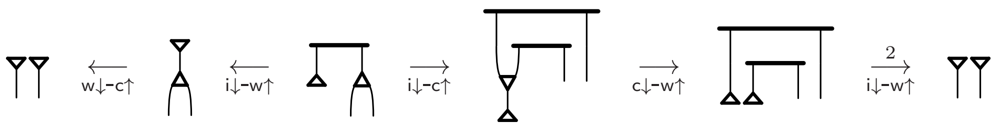
Note that every other overlapping pair can be deformed so that each rule application trivially commutes. We consider each case below.

1. 

2. 

3. 

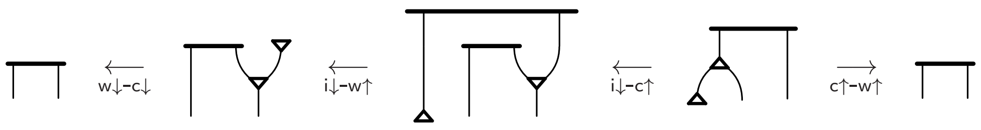
4. 

5. 

6. This case is dual to (1).

7. This case is dual to (2).

8. This case is dual to (3).

9. 

The cases where we appeal to duality follow by just flipping the indicated reductions upside down and relabelling nodes and reduction steps appropriately. □

The significance of the rewriting system norm is evident from the following results, showing that reduction steps correspond to sound manipulations of SKS-derivations.

Definition 6.7. If $\mathcal{R}$ is a binary relation on atomic flows we say that $\mathcal{R}$ *lifts polynomially* to SKS if, whenever $(fl(\Phi), \psi) \in \mathcal{R}$, we can construct a derivation Ψ in time polynomial in $|\Phi| + |\psi|$ with same premiss and conclusion as Φ and flow ψ. If f is a function on flows then we say that f lifts polynomially to SKS just if the relation $f(\cdot) = \cdot$ lifts polynomially to SKS.

In fact a derivation can always be manipulated (preserving premiss, conclusion and flow) so that it has size at most polynomial in the size of its flow, as shown in Sect. 7.2, so the dependence on size of derivation in Dfn. 6.7 is somewhat pedantic. It is only required for arguably artificial and bureaucratic derivations where there are unnecessarily long derivations in the logical fragment; such derivations exist, indeed arguably nontrivial ones, but Thm. 7.13 shows that they can be manipulated into ones of polynomial-size preserving premiss and conclusion. For this reason, we sometimes simply say that an individual flow rewrite rule is *sound* rather than saying that it lifts polynomially to SKS.

Theorem 6.8. $\underset{\mathsf{norm}}{\longrightarrow}$ *lifts polynomially to* SKS.

Proof. We assume that each norm-redex corresponds to a subderivation where the two associated rule steps are overlapping, not separated by any logical steps. This is justified since aw↓-steps permute downwards through logical steps and aw↑-steps permute upwards through logical steps, accounting for the first two rows of Fig.6-1, and ac↑-steps also permute upwards through logical steps, accounting for the third row.

In this form norm-rewrite steps are realised as follows,

$$\text{w}\!\downarrow\text{-c}\!\downarrow: \quad \mathsf{ac}\!\downarrow\dfrac{\mathsf{aw}\!\downarrow\dfrac{\bot}{a}\vee a}{a} \quad\to\quad =\dfrac{\bot\vee a}{a}$$

$$\text{i}\!\downarrow\text{-w}\!\uparrow: \quad \mathsf{ai}\!\downarrow\dfrac{\top}{\mathsf{aw}\!\uparrow\dfrac{a}{\top}\vee\bar{a}} \quad\to\quad =\dfrac{\top}{\top\vee\mathsf{aw}\!\downarrow\dfrac{\bot}{\bar{a}}}$$

$$\text{w}\!\downarrow\text{-c}\!\uparrow: \quad \mathsf{ac}\!\uparrow\dfrac{\mathsf{aw}\!\downarrow\dfrac{\bot}{a}}{a\wedge a} \quad\to\quad =\dfrac{\bot}{\mathsf{aw}\!\downarrow\dfrac{\bot}{a}\wedge\mathsf{aw}\!\downarrow\dfrac{\bot}{a}}$$

$$\text{w}\!\downarrow\text{-w}\!\uparrow: \quad \mathsf{aw}\!\uparrow\dfrac{\mathsf{aw}\!\downarrow\dfrac{\bot}{a}}{\top} \quad\to\quad \mathsf{s}\dfrac{\bot}{\top}$$

$$\text{c}\!\downarrow\text{-c}\!\uparrow: \quad \mathsf{ac}\!\uparrow\dfrac{\mathsf{ac}\!\downarrow\dfrac{a\vee a}{a}}{a\wedge a} \quad\to\quad \mathsf{m}\dfrac{\mathsf{ac}\!\uparrow\dfrac{a}{a\wedge a}\vee\mathsf{ac}\!\uparrow\dfrac{a}{a\wedge a}}{\mathsf{ac}\!\downarrow\dfrac{a\vee a}{a}\wedge\mathsf{ac}\!\downarrow\dfrac{a\vee a}{a}}$$

$$\text{i}\!\downarrow\text{-c}\!\uparrow: \quad \mathsf{ai}\!\downarrow\dfrac{\top}{\mathsf{ac}\!\uparrow\dfrac{a}{a\wedge a}\vee\bar{a}} \quad\to\quad \mathsf{ai}\!\downarrow\dfrac{\top}{\mathsf{s}\dfrac{=\dfrac{a}{a\wedge\mathsf{ai}\!\downarrow\dfrac{\top}{a\vee\bar{a}}}\vee\bar{a}}{(a\wedge a)\vee\mathsf{ac}\!\downarrow\dfrac{\bar{a}\vee\bar{a}}{\bar{a}}}}$$

and c↑-w↑ and c↓-w↑ by duality of w↓-c↓ and w↓-c↑ respectively ☐

Corollary 6.9. *Given an* SKS*-derivation* Φ *and a reduction* $fl(\Phi) = \phi_1 \underset{\mathsf{norm}}{\longrightarrow} \cdots \underset{\mathsf{norm}}{\longrightarrow}$

$\phi_n = fl(\Phi)\downarrow_{\mathsf{norm}}$ *we can construct an* SKS*-derivation with same premiss and conclusion as* Φ *and with flow* $fl(\Phi)\downarrow_{\mathsf{norm}}$ *in time polynomial in* $|\Phi| + \sum_{i=1}^{n} |\phi_i|$.

Proof. By induction on the length n of the norm-derivation. □

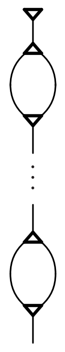
6.2 Reduction strategies

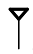
We analyse the complexity of normalising a flow under norm, presenting a class of reduction strategies that optimise the size of a $\underset{\mathsf{norm}}{\longrightarrow}$-derivation from a flow to its normal form, up to a polynomial. Our aim is to prove the following theorem:

Theorem 6.10. *The function* $\cdot\downarrow_{\mathsf{norm}}$ *mapping a flow to its unique normal form under* $\underset{\mathsf{norm}}{\longrightarrow}$ *lifts polynomially to* SKS.

The result follows from Cor. 6.9 if we can find appropriate reductions whose size are only polynomially dependent on the initial derivation and normal form of its flow.

Example 6.11. Consider the following flow,



where there are n $\mathsf{ac}\uparrow$-nodes. Notice that this flow has normal form , but the complexity of a derivation witnessing this can vary significantly. If we apply $\mathsf{c}\downarrow$-$\mathsf{c}\uparrow$-steps first then the length of a derivation to normal form is $\Theta(2^n)$, whereas applying $\{\mathsf{w}\downarrow\text{-}\mathsf{c}\downarrow, \mathsf{w}\downarrow\text{-}\mathsf{c}\uparrow\}$-steps first results in length $\Theta(n)$. These bounds follow from later results in this section, but are not difficult to prove directly.

In fact, this sort of unnecessary exponential blowup can always be avoided by applying 'weakening' rules first.

Definition 6.12. We define the following subsystems of norm:

1. $\mathsf{wk} = \{\mathsf{w}\downarrow\text{-}\mathsf{c}\downarrow, \mathsf{i}\downarrow\text{-}\mathsf{w}\uparrow, \mathsf{c}\uparrow\text{-}\mathsf{w}\uparrow, \mathsf{w}\downarrow\text{-}\mathsf{c}\uparrow, \mathsf{w}\downarrow\text{-}\mathsf{w}\uparrow, \mathsf{c}\downarrow\text{-}\mathsf{w}\uparrow\}$.
2. $\mathsf{cont} = \mathsf{norm} \setminus \mathsf{wk} = \{\mathsf{c}\downarrow\text{-}\mathsf{c}\uparrow, \mathsf{i}\downarrow\text{-}\mathsf{c}\uparrow\}$.

Proposition 6.13. *Given a flow ϕ we have:*

1. *If $\phi \underset{\mathsf{wk}}{\to} \psi$ then $|\phi| > |\psi|$.*
2. *If $\phi \underset{\mathsf{cont}}{\to} \psi$ then $|\phi| < |\psi|$.*

Lemma 6.14. *Given a flow ϕ we have $(\phi{\downarrow_{\mathsf{wk}}}){\downarrow_{\mathsf{cont}}} = \phi{\downarrow_{\mathsf{norm}}}$.*

Proof. By inspecting the rules of cont, we observe that if there are no wk-redexes in a flow ψ and $\psi \underset{\mathsf{cont}}{\to} \theta$ then there are no wk-redexes in θ. By induction on the length of a $\underset{\mathsf{cont}}{\to}$-derivation we then have that there are no wk-redexes in $(\phi{\downarrow_{\mathsf{wk}}}){\downarrow_{\mathsf{cont}}}$. But since there are also no cont-redexes, by definition of normal form, and $\mathsf{wk} \cup \mathsf{cont} = \mathsf{norm}$ we can conclude that $(\phi{\downarrow_{\mathsf{wk}}}){\downarrow_{\mathsf{cont}}}$ is already in normal form for norm. □

We can now give a proof of the main theorem of this section.

Proof of Thm. 6.10. Let Φ be an SKS-derivation with flow ϕ whose normal form under norm is ψ, and fix a derivation $\phi = \phi_1 \underset{\mathsf{wk}}{\to} \cdots \underset{\mathsf{wk}}{\to} \phi_m = \psi_1 \underset{\mathsf{cont}}{\to} \cdots \underset{\mathsf{cont}}{\to} \psi_n = \psi$, which must exist by Lemma 6.14.

By Cor. 6.9 we can construct an SKS-derivation Ψ with same premiss and conclusion as Φ and flow ψ in time polynomial in $|\Phi| + \sum_{i=1}^{m} |\phi_i| + \sum_{j=1}^{n} |\psi_i|$. However, by Prop. 6.13 we have that $|\phi_i| < |\phi_1| = |\phi| \leq |\Phi|$ for all i and $|\psi_j| < |\psi_n| = |\psi|$ for all j, and so this construction can be done in time polynomial in $(m+n)(|\Phi| + |\psi|)$.

Finally, notice also by Prop. 6.13 that each $\underset{\mathsf{wk}}{\to}$ step decreases the size of a flow, so m is bounded above by $|\phi|$ (and so also $|\Phi|$), and each cont step increases the size of a flow, so n is bounded above by $|\psi|$, whence the result follows. □

6.3 Complexity of normal forms

In this subsection we specialise previous results to flows of KS^+-proofs, reducing the complexity of norm-reduction to counting the number of certain paths in the initial flow.

Proposition 6.15. *If ϕ is the flow of a KS^+-proof, with normal form ψ under* norm*, then ψ is free of* aw↑ *and* ac↑ *nodes, i.e. contains just* KS *nodes.*

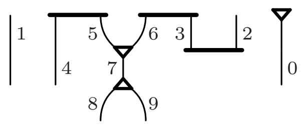
Proof. We argue by contradiction. Notice that by c↓-c↑ we can assume that all ac↑-nodes are above all ac↓-nodes in ψ, by deformation, and so a topmost one must have upper end directly connected to an aw↓ or ai↓ node, since ϕ is associated with a proof. However then ψ can be reduced by either i↓-c↑ or w↓-c↑, contradicting normality. An aw↑-node, similarly, must be above all ac↓ and ac↑ nodes by c↓-w↑ and c↑-w↑, and so must have upper end directly connected to a $\mathsf{aw}\!\downarrow$ or $\mathsf{ai}\!\downarrow$ node, and again ψ can be reduced by $\mathsf{w}\!\downarrow$-$\mathsf{w}\!\uparrow$ or $\mathsf{i}\!\downarrow$-$\mathsf{w}\!\uparrow$, contradicting normality. □

Notice that the above proposition, along with previous results in this section, allows us to transform KS^+-proofs to KS-proofs of the same conclusion. We will state this formally in Thm. 6.22 after we have determined more about the complexity of this transformation.

Definition 6.16. An ai-*path* is a (simple) path that changes (vertical) direction only at $\mathsf{ai}\!\downarrow$ and $\mathsf{ai}\!\uparrow$ nodes. We say that an ai-path is *maximal* if it cannot be extended, and that it is *open* if it begins and ends at a pending end of an edge.

The *inversion* of a path is just the same path in the reverse direction.

Example 6.17. The paths on the right, and their inversions, are exactly all the maximal ai-paths of the flow below. All of these are open except 0, since one of its ends is a $\mathsf{aw}\!\downarrow$-node. More generally all open paths are maximal but not vice-versa.



23679, 23678,
4578, 4579,
0, 1.

The following results allow us to estimate the size of the normal form of a flow, under norm, without actually constructing it.

Observation 6.18. $\underset{\mathsf{norm}}{\longrightarrow}$ *preserves the number of open* ai*-paths in a flow.*

Notation 6.19. We write $\ulcorner\phi\urcorner$ to denote the number of open ai-paths in a flow ϕ, modulo inversions, and $\#(\rho,\phi)$ to denote the number of ρ-nodes in ϕ for an atomic structural rule ρ.

Lemma 6.20. *If ϕ is the flow of a* KS*-proof then* $\ulcorner\phi\urcorner = \#(\mathsf{ai}\!\downarrow,\phi)$.

Proof. Since the flow of a proof can have no edge with upper end pending, every edge must be path-connected to an $\mathsf{aw}\!\downarrow$ or $\mathsf{ai}\!\downarrow$ node. Since no open path goes through an $\mathsf{aw}\!\downarrow$-node, and there are no $\mathsf{ai}\!\uparrow$-nodes, every open ai-path goes through a unique $\mathsf{ai}\!\downarrow$-node, and every $\mathsf{ai}\!\downarrow$-node accommodates at least one such path since there are no $\mathsf{aw}\!\uparrow$-nodes.

Now, the only other node a path can go through is an $\mathsf{ac}\!\downarrow$-node. Consequently every node in an open ai-path has in-degree 1 before passing its unique $\mathsf{ai}\!\downarrow$-node, and out-degree 1 after. Hence each $\mathsf{ai}\!\downarrow$-node accommodates exactly one open ai-path. □

Lemma 6.21. *If ϕ is the flow of a KS^+-proof, with normal form ψ under* norm*, then $|\psi| = O(|\phi| + \ulcorner \phi \urcorner)$.*

Proof. Let $\chi = \phi\downarrow_{\mathsf{wk}}$ so that $\phi \underset{\mathsf{wk}}{\overset{*}{\to}} \chi \underset{\mathsf{cont}}{\overset{*}{\to}} \psi$ by Lemma 6.14. By inspection of the rules of wk we must be able to decompose χ into two disjoint components χ_1, consisting of just ai$\downarrow$, ac$\downarrow$ and ac$\uparrow$ nodes, and χ_2 consisting of just aw$\downarrow$ and aw$\uparrow$ nodes. In fact, since ϕ was the flow of a proof, χ_2 consists of just aw$\downarrow$-nodes.

Now notice that any $\underset{\mathsf{cont}}{\to}$-derivation from χ to ψ acts only on χ_1, and so ψ can be decomposed into disjoint components ψ_1, which is the normal form of χ_1 under norm, consisting of just ai$\downarrow$ and ac$\downarrow$ nodes by Prop. 6.15, and $\psi_2 = \chi_2$, consisting of just aw$\downarrow$-nodes.

We have that $|\psi_2| = |\chi_2| \leq |\chi| \leq |\phi|$ by Prop. 6.13.

Notice that $|\psi_1| = 2 \cdot \#(\mathsf{ai}\downarrow, \psi_1) + \#(\mathsf{ac}\downarrow, \psi_1)$, since each ai$\downarrow$-node has two edges and each ac$\downarrow$-node adds a single extra edge. Since an ac$\downarrow$-node has in-degree 2 and out-degree 1, the number of ac$\downarrow$-nodes cannot outnumber the number of edges emanating from ai$\downarrow$-nodes in ψ_1, i.e. $\#(\mathsf{ac}\downarrow, \psi_1) \leq 2 \cdot \#(\mathsf{ai}\downarrow, \psi_1)$, so we have $|\psi_1| \leq 4 \cdot \#(\mathsf{ai}\downarrow, \psi_1)$. By Lemma 6.20 we then have that $|\psi_1| \leq 4 \cdot \ulcorner \psi \urcorner$, and by Obs. 6.18 that $|\psi_1| \leq 4 \cdot \ulcorner \phi \urcorner$.

Putting these together we obtain $|\psi| = |\psi_1| + |\psi_2| \leq |\phi| + 4 \cdot \ulcorner \phi \urcorner$ as required. □

Theorem 6.22. *If Φ is a KS^+-proof with flow ϕ then we can transform it to a* KS*-proof of the same conclusion in time polynomial in $|\Phi| + \ulcorner \phi \urcorner$.*

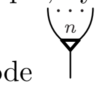
Proof. Let ψ be the normal form of ϕ under norm. By Prop. 6.15 we have that ψ contains just KS-nodes, and so by Thm. 6.10 we have that a KS-proof with same conclusion as Φ can be constructed in time polynomial in $|\Phi| + |\psi|$. The statement follows by the bound $|\psi| = O(|\phi| + \ulcorner \phi \urcorner) = O(|\Phi| + \ulcorner \phi \urcorner)$ given by Lemma 6.21. □

6.4 Estimating the number of paths in a flow

It is not difficult to see that the main contributor to an increase of flow size reducing under norm is the rule c$\downarrow$-c$\uparrow$. It can sometimes cause an exponential blowup, as evident in Ex. 6.11 if there were no aw$\downarrow$-node at the top.

The following provides a simple estimate of the number of paths in a flow, and so the complexity of flow normalisation under norm, which are used to obtain short KS-proofs of various combinatorial principles in Chapt. 9.

Definition 6.23 (Dimensions of a flow)**.** The *length* of a flow is the maximum number of times the type of node changes in a (vertically) directed path. The *width* of a flow is the maximum size of a connected subflow containing just one type of node. The *breadth* of a flow is the number of connected components it has.

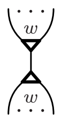
The above definition is perhaps most easily understood by allowing $\mathsf{ac}\downarrow$ and $\mathsf{ac}\uparrow$ nodes to have unbounded in-degree and out-degree respectively. For example, by just collapsing any configuration of $n-1$ connected $\mathsf{ac}\downarrow$-nodes to a single node  and similarly for configurations of $\mathsf{ac}\uparrow$-nodes. We can then consider the length of the flow to be just the maximum length, in the usual sense, of a vertically directed path, and the width to be the maximum out-degree or in-degree of a node.

Notice that replacing $\mathsf{ac}\downarrow$ and $\mathsf{ac}\uparrow$ nodes with these 'super' nodes is sound, since we can permute $\mathsf{ac}\uparrow$-steps upwards and $\mathsf{ac}\downarrow$-steps downwards in a derivation as necessary, and this does not affect the size of a flow or derivation superpolynomially. In particular the number of maximal paths is preserved.

Proposition 6.24. *If ϕ is a flow consisting of just KS^+-nodes with width w, length l and breadth b, then $\ulcorner\phi\urcorner = b \cdot w^{l+O(1)}$.*

Proof. It suffices to consider the case when $b = 1$, since paths in different connected components are disjoint, and that ϕ is free of $\{\mathsf{aw}\downarrow, \mathsf{aw}\uparrow\}$-nodes, since replacing those nodes with pending edges can only increase the number of open ai-paths, while not changing its dimensions.

Assuming that there are no $\mathsf{ai}\downarrow$-nodes in ϕ, in the worst case we just have a sequence of configurations,



in series vertically, and each configuration multiplies the number of paths by w and adds 2 to the length, yielding the bound $\ulcorner\phi\urcorner = w^{\frac{l}{2}+O(1)}$.

Now any ai-path goes through at most one $\mathsf{ai}\downarrow$-node, so we have that the number of ai-paths going through any $\mathsf{ai}\downarrow$-node is the product of the number of vertically directed paths going through each of its edges. Since adding $\mathsf{ai}\downarrow$-nodes to a flow does not change its length, and the number of $\mathsf{ai}\downarrow$-nodes in a connected component is bounded by its width, from the above bound on $\mathsf{ai}\downarrow$-free flows we obtain that $\ulcorner\phi\urcorner = w^{l+O(1)}$, as required. □

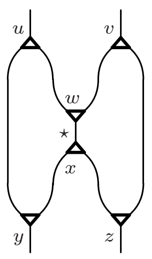
Remark 6.25. We generally use the trivial upper bound of size of flow for width and breadth, yielding the estimate $\ulcorner\phi\urcorner = |\phi|^{l+O(1)}$ for a KS^+-flow ϕ.

6.5 More sophisticated estimate of number of paths

Sometimes measuring the dimensions of a flow gives an insufficiently accurate estimate of its number of paths. This section shows that only certain interactions between $\mathsf{ac}\!\downarrow$ and $\mathsf{ac}\!\uparrow$ nodes generate significant complexity in a KS^+-flow. In the absence of these the number of open ai-paths in a flow is still polynomial in its size, regardless of its length.

Definition 6.26. A *contraction loop* in a flow is a pair of $(\mathsf{ac}\!\uparrow, \mathsf{ac}\!\downarrow)$ nodes (ν_1, ν_2) such that there are at least two disjoint (directed) paths between ν_1 and ν_2

For example we give the following flow and all its contraction loops,



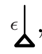
$(u, y), (v, z).$

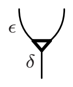
whereas every other pair has only one path between them. If the edge $\star$ were broken and there was no path from w to x then there would be no contraction loops at all.

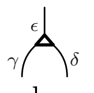
Lemma 6.27. *If there are no contraction loops in a KS^+-flow ϕ then $\ulcorner \phi \urcorner \leq |\phi|^3$.*

Proof. For an edge ϵ in ϕ consider the following two notions:

- The *weight* of ϵ, denoted $w(\epsilon)$, is the number of directed paths from ϵ to the bottom of ϕ, i.e. to an $\mathsf{aw}\!\uparrow$-node or an edge with lower end pending.
- For an atomic structural rule ρ let $N(\rho, \epsilon)$ denote the number of ρ-nodes below ϵ that are connected to ϵ by a directed path.

We show that $w(\epsilon) \leq N(\mathsf{aw}\!\uparrow, \epsilon) + N(\mathsf{ac}\!\uparrow, \epsilon) + 1$ by induction on the distance of ϵ from the bottom of ϕ. The inequality is clear for the base cases when ϵ is directly connected to a $\mathsf{aw}\!\uparrow$ node, , and when ϵ has lower end pending, $|$. We have two inductive steps:

1. ϵ is an upper edge of an $\mathsf{ac}\!\downarrow$-node, . In this case we clearly have that $w(\epsilon) = w(\delta)$ and so the inequality follows by the inductive hypothesis.

2. ϵ is the upper edge of an $\mathsf{ac}\!\uparrow$-node, . Observe that, since there are no contraction loops in ϕ and $\mathsf{ac}\!\downarrow$ is the only node type with in-degree greater than

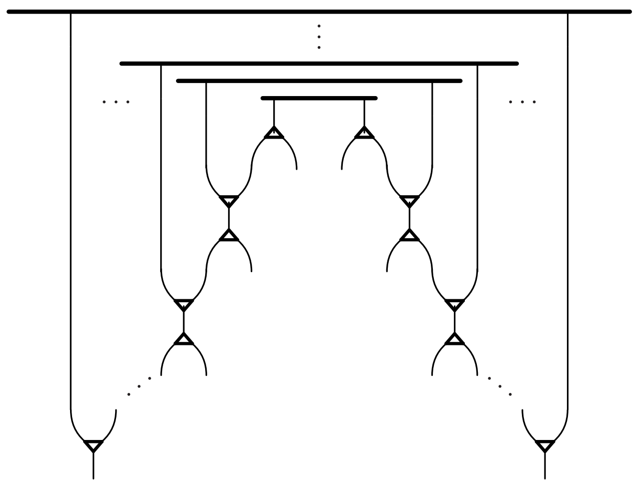
1, any node below this $\mathsf{ac}\!\uparrow$-node can be directed-path-connected to at most one of γ or δ. Consequently we have that $N(\mathsf{aw}\!\uparrow, \epsilon) = N(\mathsf{aw}\!\uparrow, \gamma) + N(\mathsf{aw}\!\uparrow, \delta)$ and $N(\mathsf{ac}\!\uparrow, \epsilon) = N(\mathsf{ac}\!\uparrow, \delta) + N(\mathsf{ac}\!\uparrow, \gamma) + 1$. Therefore,

$$\begin{aligned}
w(\epsilon) &= w(\delta) + w(\gamma) \\
&\leq (N(\mathsf{aw}\!\uparrow, \delta) + N(\mathsf{ac}\!\uparrow, \delta) + 1) + (N(\mathsf{aw}\!\uparrow, \gamma) + N(\mathsf{ac}\!\uparrow, \gamma) + 1) \\
&\leq (N(\mathsf{aw}\!\uparrow, \delta) + N(\mathsf{aw}\!\uparrow, \gamma) + (N(\mathsf{ac}\!\uparrow, \delta) + N(\mathsf{ac}\!\uparrow, \gamma) + 1) + 1 \\
&\leq N(\mathsf{aw}\!\uparrow, \epsilon) + N(\mathsf{ac}\!\uparrow, \epsilon) + 1
\end{aligned}$$

Notice in particular that by this inequality we have that $w(\epsilon) \leq |\phi|$.

Clearly the number of open ai-paths going through an edge with upper end pending is bounded above by its weight, and so by $|\phi|$ by the bound above, while the number of open ai-paths going through any $\mathsf{ai}\!\downarrow$ node is bounded above by the product of the weights of each of its edges, and so by $|\phi|^2$. In particular we have that the number of open ai-paths going through any edge at the top of a flow is bounded above by $|\phi|^2$, and there are at most $|\phi|$ many such edges, whence the bound follows. $\square$

Example 6.28. In fact the bound given above is optimal, up to multiplication by a constant. Consider the flow ϕ below, where there are n $\mathsf{ai}\!\downarrow$ nodes:



Clearly the flow has size linear in n and notice that the weights (as defined in the above proof) of the topmost edges, starting from the left, are $1, 2, \ldots, n, n, \ldots, 1$ respectively. Consequently the number of open ai-paths going through the $\mathsf{ai}\!\downarrow$ nodes, starting from the outside, is $1^2, 2^2, \ldots, n^2$ respectively, and so taking the sum we obtain $\ulcorner \phi \urcorner = \Omega(n^3)$.

Notice also that ϕ has length linear in n and width 2, and so the upper bound on $\ulcorner \phi \urcorner$ given by Prop. 6.24 is exponential, considerably worse than the bound given in

Lemma 6.27.

Chapter 7

Complexity of the logical fragment of deep inference

In this chapter we turn to the logical fragment of SKS. The objects of study here are $\{\mathsf{s}, \mathsf{m}\}$-derivations, and we analyse them in the setting of term rewriting. The motivation is to understand the contribution of these rules to the complexity of a derivation; as we will see, $\{\mathsf{s}, \mathsf{m}\}$ generally does not contribute superpolynomially to the size of a derivation, but it is expressive enough to encode all the information of a proof, from the point of view of complexity, and consequently determines the complexity of proof search in deep inference.

As we have mentioned before, the normalisation and complexity results of the previous chapter hold independently of the rules of the logical fragment, provided all logical rules are linear and switch and medial are derivable - derivability of both these rules is necessary in order to obtain atomicity for the structural rules. Consequently it is an interesting pursuit to see what effects other linear rules may have on a proof system, and we consider examples of such rules also in this chapter.

One thing to note is that the polynomial bounds we obtain on lengths of linear derivations are specific to $\{\mathsf{s}, \mathsf{m}\}$-derivations; there is nothing to stop other linear rules from contributing superpolynomially to the size of a derivation, again something that we will see in this chapter.

The material in this chapter is based on work that has appeared in [Das13].

7.1 The term rewriting setting

Term rewriting and deep inference have a lot in common, indeed one can just view deep inference systems as term rewriting systems on propositional logic, with CoS derivations corresponding to rewrite paths. In this chapter we find that many questions about the complexity of the logical fragment of deep inference, or indeed of proofs in general, can be phrased as natural questions in the setting of term rewriting.

On a point of notation, to maintain consistency with the term rewriting literature, we now use formula variables A, B, etc. as formal symbols occurring in terms s, t etc., and we denote ground terms, i.e. formulae, by meta-variables α, β etc.

As convention, we construe inference rules as term rewriting rules. We no longer assume that $=$ is contained in every system, but rather distinguish between the rebracketing rules and the unit rules. We denote by AC the equational theory generated by the rebracketing rules, and by U the equational theory generated by the unit rules.

We associate deep inference derivations with rewrite paths by their translation to CoS form.

We define the system $\mathsf{MS} = \{\mathsf{s}, \mathsf{m}\}$ and the system $\mathsf{MSU} = \mathsf{MS} \cup \mathsf{U}$. All systems are assumed to operate modulo AC, although we may indicate an instance of AC, or any other equational theory we are working modulo, by a 'fake' inference step $\dfrac{A}{B}$.

7.2 Length of paths with units

In this section we address the complexity of rewriting paths in MSU. The length of MS-paths is well-known to be polynomial, and we give a simple proof below that the length is at most cubic in the size of an input term. Much tighter bounds can be obtained, and this is the subject of ongoing work by Bruscoli, Guglielmi and Straßburger.[1]

It should be pointed out that the general belief that units do not contribute to the complexity of a proof is commonplace in the deep inference community, with some results as folklore, for example the theorem below. Nonetheless, the technicalities of proving this belief, or even formalising what this means, seems nontrivial to me and this sentiment is communicated via numerous examples.

Theorem 7.1. MS *has only polynomial-length paths.*

Proof. Let $n(t)$ denote the number of $\wedge$s occurring in a term t, and let $m(t)$ denote the number of pairs of leaves in the term-tree of t whose least common connective is $\wedge$. Clearly each medial step reduces the n-value of a term while each switch step reduces the m-value of a term, not changing the n-value.

Therefore each MS-step strictly decreases the lexicographical product $n \times m$. Since n is linear in the size of a term and m is quadratic, we have that an MS-path can contain at most a cubic number of steps. $\square$

[1] Personal correspondence.

The situation becomes more complicated when units are considered. Since the rules of U are bidirectional, cycles can be trivially constructed, yielding infinite rewrite paths. Moreover non-cyclic infinite 'increasing' paths can be constructed:

$$a \quad\to\quad \top \wedge a \quad\to\quad \top \wedge \top \wedge a \quad\to\quad \top \wedge \top \wedge \top \wedge a \quad\to\quad \cdots$$

One approach here would be to conduct rewriting modulo the equational theory generated by U, i.e. consider formulae equivalent up to U-rewriting.[2]

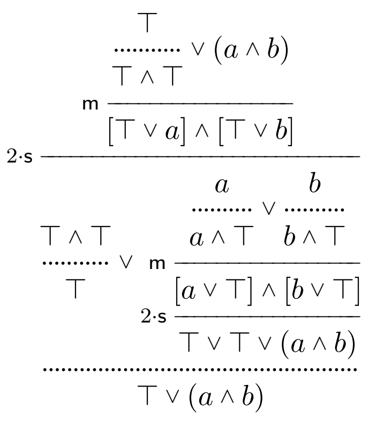
Definition 7.2 (Rewriting modulo)**.** Let $\mathcal{R}$ be a rewriting system and $\sim$ an equivalence relation on the terms of $\mathcal{R}$. A derivation in $\mathcal{R}/\sim$ is a sequence,

$$s \sim s_1 \to t_1 \sim s_2 \to t_2 \sim \cdots \to t_k \sim t$$

where each $s_i \to t_i$ is a one-step rewrite in $\mathcal{R}$ and $s_i \not\sim t_i$.

We should note that this is a nonstandard definition of rewriting modulo, since we enforce that each rewriting step is between $\sim$-distinct terms. This condition crucially affects termination of a system, but makes sense in the current setting since the equivalence relations induced by our equations can be checked efficiently.

In any case this approach does not quite work here, since we can still construct cycles when rewriting modulo U. For example the following,



is a derivation for a cycle $\top \vee (a \wedge b) \to \cdots \to \top \vee a \vee b \to \cdots \to \top \vee (a \wedge b)$.

These situations only occur when a subterm appears in conjunction with $\bot$ or disjunction with $\top$, a concept we later define as *trivialisation.* They can be avoided by

[2]We will not address here complexity issues arising from such an approach. There are ways to present such rewritings such that each step can still be checked efficiently [BG09b].

adding to U the following 'non-linear' equations:

$$A \vee \top = \top \qquad A \wedge \bot = \bot$$

Let us call the resulting system U'. We will need the following two results in order to deduce termination.

Proposition 7.3. *If two unit-free formulae are distinct, modulo associativity and commutativity, with each propositional variable occurring at most once, then they compute distinct boolean functions.*

Proof. See e.g. [Gur77]. □

Proposition 7.4. *Every term is* U'*-equivalent to a unique unit-free term or* $\top$ *or* $\bot$.

Proof. By inspection of U', each equation has a direction that eliminates a unit occurrence, and the equations cover every context in which a unit might occur, except the empty context. Assigning this direction to U' therefore yields a terminating system whose normal forms are unit-free or a single unit.

Uniqueness follows by Prop. 7.3 above, along with the observation that U' preserves the boolean function computed by a formula. □

From these we can deduce the strong normalisation property.

Theorem 7.5. *Rewriting in* MS/U' *is terminating.*

Proof. Without loss of generality, assume the input is a formula (i.e. a ground term), since MS and U' do not distinguish between atoms and variables, and that no variable occurs more than once, since no rule of the system duplicates variables in either direction. By Props. 7.4 and 7.3 it follows that, for each step $\alpha \to \beta$ in a MS/U'-derivation, α and β compute distinct boolean functions, otherwise they are in the same U'-equivalence class.

There are 2^n assignments on n atoms, and each boolean function determines a unique set of these assignments. Since rewriting in MS/U' is sound, any rewrite path determines a strictly increasing sequence of sets of assignments with respect to $\subset$. □

7.2.1 An exponential-length path in MS/U'

Notice that the complexity bound on termination extracted from the proof of Thm. 7.5 is exponential, unlike the unit-free case which is polynomial. Perhaps surprisingly, one cannot do better than this, and we prove this by constructing explicit rewrite-paths of exponential length.

We present a new class of rules, collectively known as *supermix*, that are derivable in MSU and show that one can construct exponential-length paths with it, with exponentially many U'-distinct formulae occurring.

Definition 7.6 (Supermix)**.** We define the supermix rules, indexed by n, below:

$$\mathsf{smix} \quad : \quad A \wedge \bigvee_{i=1}^{n} B_i \quad \to \quad A \vee \bigwedge_{i=1}^{n} B_i$$

Each supermix rule is clearly a sound linear inference and, for the special case when $n = 1$, it coincides with the usual mix rule, $A \wedge B \to A \vee B$.

The following results prove that supermix is derivable in MSU. Recall Rmk. 3.15 where we gave derivations from $\bot$ to $\top$ for both s/U and m/U. We will simply write $\dfrac{\bot}{\top}$ as shorthand for one of these derivations.

Lemma 7.7. *There are* MS/U*-paths from* $\bigvee\limits_{i=1}^{n} B_i$ *to* $\top \vee \bigwedge\limits_{i=1}^{n} B_i$.

Proof. We proceed by induction on n.

Base case: When $n = 1$ we have $\dfrac{\bot}{\top} \vee B_1$.

Inductive step: Suppose there are such derivations Φ_r for $r < n$. Define:

$$\Phi_n \quad \equiv \quad \mathsf{2{\cdot}s}\,\dfrac{\mathsf{m}\,\dfrac{\dfrac{B_n}{\top \wedge B_n} \vee \dfrac{\begin{array}{c} \bigvee\limits_{i=1}^{n-1} B_i \\ \Phi_{n-1} \Big\Vert \mathsf{MSU} \\ \top \vee \bigwedge\limits_{i=1}^{n-1} B_i \end{array}}{\top \wedge \left[\top \vee \bigwedge\limits_{i=1}^{n-1} B_i\right]}}{[\top \vee B_n] \wedge \left[\top \vee \top \vee \bigwedge\limits_{i=1}^{n-1} B_i\right]}}{\dfrac{\top \vee \top \vee \top}{\top} \vee \left(B_n \wedge \bigwedge\limits_{i=1}^{n-1} B_i\right)}$$

□

Theorem 7.8. smix *is derivable in* MS/U.

Proof. Let Φ_n be the derivations constructed in Lemma 7.7 above. The derivation is as follows:

$$\mathsf{s}\,\frac{A \wedge \left[\begin{array}{c} \bigvee_{i=1}^{n} B_i \\ \Phi_n \Big\| \mathsf{MSU} \\ \top \vee \bigwedge_{i=1}^{n} B_i \end{array}\right]}{\left[\begin{array}{c} A \wedge \top \\ \vdots \\ A \end{array}\right] \vee \bigwedge_{i=1}^{n} B_i}$$

$\square$

Note that the premiss and conclusion of a supermix step are distinct modulo U', since they are unit-free and compute distinct boolean functions, and so we can construct an exponential-length path in MS/U' as follows:

$$\Lambda_1 \equiv a_1 \qquad , \qquad \Lambda_{n+1} \equiv \quad \mathsf{smix}\,\frac{a_{n+1} \wedge \left[\begin{array}{c} \bigwedge_{i=1}^{n} a_i \\ \Lambda_n \Big\| \mathsf{smix} \\ \bigvee_{i=1}^{n} a_i \end{array}\right]}{a_{n+1} \vee \left[\begin{array}{c} \bigwedge_{i=1}^{n} a_i \\ \Lambda_n \Big\| \mathsf{smix} \\ \bigvee_{i=1}^{n} a_i \end{array}\right]}$$

Denoting the length of Λ_n by l_n, we have that $l_1 = 0$ and $l_n = 2 \cdot l_{n-1} + 1$, so $l_n = \sum_{i=0}^{n-2} 2^i = 2^{n-1} - 1$.

In fact these rewrite paths exhibit the maximum possible length of an MS/U'-derivation. Notice that any formula in which each atom occurs at most once satisfies an odd number of assignments,[3] of which there are precisely 2^{n-1} over n variables, and so there can be no path over n variables of length greater than $2^{n-1} - 1$.

7.2.2 Construction of polynomial-length paths

The cause of problems in the (complexity of) termination of MSU seems to be the trivialising of atoms and variables in a derivation, by putting them in disjunction

[3]Let $\|A\|$ denote the number of assignments satisfied by A. Clearly $\|a_1\| = 1$ is odd, and notice that $\|A \wedge B\| = \|A\| \cdot \|B\|$ and $\|A \vee B\| = 2^{|B|} \cdot \|A\| + 2^{|A|} \cdot \|B\| - \|A\| \cdot \|B\|$, which are odd if both $\|A\|$ and $\|B\|$ are odd.

with $\top$ or conjunction with $\bot$. We say that such atoms are *trivialised* and show that, although there are paths of exponential length, any two terms with a MSU-path between them has one of polynomial length. The general idea is to 'push' trivialised atoms and variables to one side and reduce to the unit-free case, before reintroducing the trivialised symbols.

Throughout this section we use 'dotted' steps $\begin{array}{c} s \\ \cdots \\ t \end{array}$ to denote U-steps in a derivation, to help distinguish MS-steps from U-steps. This is technically an overloading of notation, but does not cause any problem since there is a polynomial-size MSU-derivation from s to t just if there is a polynomial-size MS/U-derivation from s to t.

We will make use of Lemma 3.11 in this subsection, as well as the following analogue result for medial.

Lemma 7.9. *There are polynomial-size derivations* $\begin{array}{c}(\bot \wedge A) \vee \xi\{\bot\} \\ \| \mathsf{m}\cup\mathsf{U} \\ \xi\{\bot \wedge A\}\end{array}$ *and* $\begin{array}{c}\xi\{\top \vee A\} \\ \| \mathsf{m}\cup\mathsf{U} \\ [\top \vee A] \wedge \xi\{\top\}\end{array}$.

Proof. We proceed by induction on the depth of the hole in $\xi\{\;\}$. The base cases are trivial, and we give the inductive steps for the first derivation below,

$$\mathsf{m}\,\frac{\left(\begin{array}{c}\bot \\ \cdots\cdots \\ \bot \wedge \bot\end{array} \wedge A\right) \vee (\xi\{\bot\} \wedge B)}{\begin{array}{c}(\bot \wedge A) \vee \xi\{\bot\} \\ {\scriptstyle IH}\big\| \mathsf{m} \\ \xi\{\bot \wedge A\}\end{array} \wedge \begin{array}{c}\bot \vee B \\ \cdots\cdots \\ B\end{array}} \quad , \qquad \begin{array}{c}(\bot \wedge A) \vee [\xi\{\bot\} \vee B] \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ (\bot \wedge A) \vee \xi\{\bot\} \vee B \\ {\scriptstyle IH}\big\| \mathsf{m} \\ \xi\{\bot \wedge A\}\end{array}$$

where derivations marked IH are obtained by the inductive hypothesis. The second derivation is obtained by duality. $\square$

Definition 7.10 (Trivialisation)**.** A term is *trivial* if it is a disjunction containing $\top$ or a conjunction containing $\bot$. In a derivation or rewrite path we say that an atom or variable is *trivialised* if at any point it occurs inside a trivial subterm.

Lemma 7.11. *If* $\begin{array}{c}\xi\{A\} \\ \Phi\| \mathsf{MSU} \\ \zeta\{A\}\end{array}$ *is a derivation where A occurs trivialised then there is a derivation* $\begin{array}{c}\xi\{\top \vee A\} \\ \| \mathsf{MSU} \\ \zeta\{\bot \wedge A\}\end{array}$ *of size* $|\Phi|^{O(1)}$.

Proof. There are two cases. In the first case we transform the derivation as follows,

$$\begin{array}{c} \xi\{A\} \\ \Phi \Vert \mathsf{MSU} \\ \eta\{\top \vee \zeta\{A\}\} \\ \Psi \Vert \mathsf{MSU} \\ \xi'\{A\} \end{array} \quad \to \quad \begin{array}{c} \xi\{\top \vee A\} \\ \Phi' \Vert \mathsf{MSU} \\ \eta\left\{\begin{array}{c} \top \vee \zeta\left\{\begin{array}{c} \top \vee \ \mathsf{s}\,\dfrac{\begin{array}{c} A \\ \hdashline [\top \vee \bot] \wedge A \end{array}}{\top \vee (\bot \wedge A)} \\ \hdashline \top \vee (\bot \wedge A) \end{array}\right\} \\ \bullet \Vert \mathsf{s} \\ \top \vee \zeta\{\bot \wedge A\} \\ \hdashline \top \vee \zeta\{\bot \wedge A\} \end{array}\right\} \\ \Psi' \Vert \mathsf{MSU} \\ \xi'\{\bot \wedge A\} \end{array}$$

where Φ' and Ψ' are obtained by substituting $\top \vee A$ and $\bot \wedge A$ respectively everywhere for A, and the derivation marked $\bullet$ is obtained by Lemma 3.11. In the second case we transform the derivation as follows,

$$\begin{array}{c} \xi\{A\} \\ \Phi \Vert \mathsf{MSU} \\ \eta\{\bot \wedge \zeta\{A\}\} \\ \Psi \Vert \mathsf{MSU} \\ \xi'\{A\} \end{array} \quad \to \quad \begin{array}{c} \xi\{\top \vee A\} \\ \Phi' \Vert \mathsf{MSU} \\ \eta\left\{\begin{array}{c} \begin{array}{c} \bot \\ \hdashline \bot \wedge \bot \end{array} \wedge \ \zeta\left\{\begin{array}{c} \top \vee \ \mathsf{s}\,\dfrac{\begin{array}{c} A \\ \hdashline [\top \vee \bot] \wedge A \end{array}}{\top \vee (\bot \wedge A)} \\ \hdashline \top \vee (\bot \wedge A) \end{array}\right\} \\ \bullet \Vert \mathsf{s} \\ 2{\cdot}\mathsf{s}\,\dfrac{\top \vee \zeta\{\bot \wedge A\}}{\begin{array}{c} (\bot \wedge \top) \vee (\bot \wedge \zeta\{A\}) \\ \hdashline \bot \wedge \zeta\{\bot \wedge A\} \end{array}} \end{array}\right\} \\ \Psi' \Vert \mathsf{MSU} \\ \zeta\{\bot \wedge A\} \end{array}$$

where Φ' and Ψ' are obtained by substituting $\top \vee A$ and $\bot \wedge A$ respectively everywhere for A, and the derivation marked $\bullet$ is obtained by Lemma 3.11. ☐

Lemma 7.12. *An* MSU*-path where no atoms or variables occur trivialised can be transformed into an* MS*-path with* U*-equivalent premiss and conclusion.*

Proof. We simply reduce every line in a MSU-derivation to a unit-free term by U. Since no atoms or variables are trivialised we do not need any rules of $\mathsf{U}' \setminus \mathsf{U}$. We rewrite derivations using the four possible cases below:

$$\mathsf{s}\frac{s \wedge [\bot \vee t]}{(s \wedge t) \vee \bot} \quad \to \quad s \wedge t \qquad\qquad \mathsf{s}\frac{\top \wedge [s \vee t]}{(\top \wedge s) \vee t} \quad \to \quad s \vee t$$

$$\mathsf{m}\frac{(s \wedge t) \vee (\bot \wedge \bot)}{[s \vee \bot] \wedge [t \vee \bot]} \quad \to \quad s \wedge t \qquad\qquad \mathsf{m}\frac{(s \wedge \top) \vee (t \wedge \top)}{[s \vee t] \wedge [\top \vee \top]} \quad \to \quad s \vee t$$

Any other combination of rules with units results in some term in either the premiss or conclusion being trivialised. □

Theorem 7.13. *Every* MSU*-path can be transformed into one with same premiss and conclusion and whose size is polynomial in the size of its premiss and conclusion.*

Proof. Let Φ be an MSU-derivation. If there are no trivialisations then transform it into an MS-derivation by Lemma 7.12 which must be of polynomial size by Thm. 7.1.

Otherwise assume there is a trivialised variable in Φ, say A_1, and transform Φ as follows:

$$\begin{array}{c} \xi\{A_1\} \\ \Phi \Big\Vert \mathsf{MSU} \\ \zeta\{A_1\} \end{array} \quad \to \quad \begin{array}{c} \xi\{\top \vee A_1\} \\ \Phi' \Big\Vert \mathsf{MSU} \\ \zeta\{\bot \wedge A_1\} \end{array} \quad \to \quad \begin{array}{c} \xi\{\top \vee \bot\} \\ \Phi_1 \Big\Vert \mathsf{MSU} \\ \zeta\{\bot \wedge \bot\} \end{array}$$

where Φ' is obtained from Φ by Lemma 7.11 and Φ_1 from Φ' by substituting $\bot$ for every instance of A_1.

Now do the same for Φ_1, and repeat this process until either there are no trivialisations in some Φ_k. (Note that it is not sufficient to just do all the trivialised variables at once, since the transformation above may result in new trivialisations.)

Now by Lemma 7.12 we can transform Φ_k into an MS-derivation Ψ, with same premiss and conclusion modulo U, which we can assume has polynomial size by Thm. 7.1.

$$\begin{array}{c} \xi\{\top \vee \bot\} \cdots \{\top \vee \bot\} \\ \Phi_k \Big\Vert \mathsf{MSU} \\ \zeta\{\bot \wedge \bot\} \cdots \{\bot \wedge \bot\} \end{array} \quad \to \quad \begin{array}{c} \xi\{\top \vee \bot\} \cdots \{\top \vee \bot\} \\ \cdots\cdots\cdots\cdots\cdots \\ s \\ \Psi \Vert \mathsf{MS} \\ t \\ \cdots\cdots\cdots\cdots\cdots \\ \zeta\{\bot \wedge \bot\} \cdots \{\bot \wedge \bot\} \end{array}$$

The complete transformation is as follows,

$$\xi\{A_1\}\cdots\{A_k\} \atop {\Phi\Vert\mathsf{MSU} \atop \zeta\{A_1\}\cdots\{A_k\}} \quad \rightarrow \quad \begin{array}{c} \xi\left\{\begin{array}{c} A_1 \\ \hdashline [\top\vee\bot]\wedge A_1 \\ \hline \mathsf{s}\quad \top\vee(\bot\wedge A_1) \end{array}\right\}\cdots\left\{\begin{array}{c} A_k \\ \hdashline [\top\vee\bot]\wedge A_k \\ \hline \mathsf{s}\quad \top\vee(\bot\wedge A_k) \end{array}\right\} \\ \circ\Vert\mathsf{s} \\ \left[\begin{array}{c} \xi\{\top\vee\bot\}\cdots\{\top\vee\bot\} \\ \hdashline s \\ \Psi\Vert\mathsf{MS} \\ t \\ \hdashline \zeta\{\bot\wedge\bot\}\cdots\{\bot\wedge\bot\} \end{array} \vee(\bot\wedge A_1)\vee\cdots\vee(\bot\wedge A_k)\right] \\ \bullet\Vert\mathsf{m} \\ \zeta\left\{\begin{array}{c} \dfrac{\bot}{\top}\wedge A_1 \\ \hdashline A_1 \end{array}\right\}\cdots\left\{\begin{array}{c} \dfrac{\bot}{\top}\wedge A_k \\ \hdashline A_k \end{array}\right\} \end{array}$$

where the derivations marked $\circ, \bullet$ are obtained by repeatedly applying Lemma 7.9. $\square$

Remark 7.14. By the above theorem it follows that any derivation can be transformed to one with the same premiss and conclusion, the same atomic flow and whose size is polynomial in the size of its atomic flow. Consequently the normalisation procedures of Chapt. 6, along with the theorem above, yield derivations of size polynomial in the size and number of open ai-paths of the flow of the input derivation.

7.3 Complexity of characterising MS

The motivation behind this section originates from the following result in [Str07].

Theorem 7.15 (Straßburger)**.** *There are polynomial-time criteria deciding whether there is a* s *or* m *rewrite path between two terms.*

In the same work the task of characterising MS was raised as an open problem.

In this section we give a polynomial-time reduction from the problem of finding a Hilbert-Frege proof of a given tautology of a given size to the problem of finding a MS-rewrite path between two terms. Consequently, we deduce that there is no polynomial-time characterisation of MS (and also MSU) under the assumption that integer factoring cannot be computed by polynomial-size circuits, using a result of Bonet et al. on the automatisability of Hilbert-Frege systems.

7.3.1 Reducing proof-search to rewriting in MS

Throughout this section, for a formula α, we write α^n to denote $\overbrace{\alpha \wedge \cdots \wedge \alpha}^{n}$ and $n \cdot \alpha$ to denote $\overbrace{\alpha \vee \cdots \vee \alpha}^{n}$.

The goal of this section is to prove the following result.

Theorem 7.16. *A Hilbert-Frege proof π of a tautology τ can be polynomially transformed into a unit-free derivation of the following form,*

$$\begin{array}{c} \bigwedge_i [a_i \vee \bar{a}_i]^{r_i} \\ \Big\| \mathsf{MS} \\ \tau' \end{array}$$

where τ' is obtained from $\sigma = \tau \vee (a_1 \wedge \bar{a}_1) \vee \cdots \vee (a_n \wedge \bar{a}_n)$, where a_i are the propositional variables occurring in τ, by,

- *replacing each occurrence of a_i by $k \cdot m_i \cdot a_i$, where m_i is the number of occurrences of $\bar{a}_i$ in σ,*
- *replacing each occurrence of $\bar{a}_i$ by $k \cdot n_i \cdot \bar{a}_i$, where n_i is the number of occurrrences of a_i in σ,*

where $k = O(|\pi|)$ and r_i is determined by m_i, n_i and k by linearity of MS.

We sketch the basic ideas of the proof, avoiding heavy syntax, from which it should be easy to reconstruct a fully detailed argument. We will require several intermediate results, in which we deal with formulae rather than general terms for simplicity, and we assume that all formulae in the conclusions of proofs are unit-free unless otherwise stated.

Proposition 7.17. *A KS-proof Φ of a tautology τ can be polynomially transformed to a unit-free derivation of the following shape,*

$$\begin{array}{c} \bigwedge_i a_i \vee \bar{a}_i \vee \beta_i \\ \Big\| \mathsf{MS} \\ \tau' \end{array}$$

where τ' differs from τ only by replacing atom occurrences a by a disjunction $k \cdot a$ for some $k \leq |\Phi|$.

Proof sketch. First replace every trivialised atom occurrence by $\top$ and alter affected steps as in Prop. 3.6, repeating as necessary if there are further trivialisations. This procedure must terminate in linear time since the number of atom occurrences is strictly decreasing at each step, yielding a KS-proof of τ with no trivial atom occurrences.

Now permute $\mathsf{ac}\!\downarrow$-steps downwards to the bottom and $\mathsf{ai}\!\downarrow$ and $\mathsf{aw}\!\downarrow$ steps upwards to the top; this does not create any new trivialisations as permuting $\mathsf{ai}\!\downarrow$ or $\mathsf{aw}\!\downarrow$ upwards only removes some unit occurrences. Let k be the maximum number of $\mathsf{ac}\!\downarrow$-steps immediately above an atom in the conclusion and introduce $\mathsf{aw}\!\downarrow$-steps at the bottom to ensure that every atom occurs with multiplicity k; again permute $\mathsf{aw}\!\downarrow$-steps to the top.

At this point reduce every line in the proof by U to a unit-free formula, altering affected steps as in Lemma 7.12 and apply Lemma 3.11 at the top to obtain a premiss in conjunctive normal form. $\square$

Lemma 7.18. *Given a formula* $\bigwedge_i a_i \vee \bar{a}_i \vee \beta_i$ *there is a polynomial-size derivation of the following form,*

$$\begin{array}{c} \alpha \\ \Big\Vert \mathsf{s} \\ \bigwedge_i k \cdot a_i \vee k \cdot \bar{a}_i \vee \beta_i \end{array}$$

where α *is a valid formula in conjunctive normal form, for some* $k \leq \max_i |\beta_i|$.

Proof. Freely apply the inverse of s to each β_i to obtain a formula β'_i of same size in conjunctive normal form, with disjunctions $\beta'_{i1}, \ldots, \beta'_{is}$. Construct the following derivations,

$$\begin{array}{c} [a_i \vee \bar{a}_i \vee \beta'_{i1}] \wedge \cdots \wedge [a_i \vee \bar{a}_i \vee \beta'_{is}] \\ \Big\Vert \mathsf{s} \\ \left[k_i \cdot a_i \vee k_i \cdot \bar{a}_i \vee \begin{array}{c} \beta'_i \\ \Vert \mathsf{s} \\ \beta_i \end{array} \right] \end{array}$$

where k_i is the number of conjuncts in β'_i, and choose $k = \max_i k_i$. Validity follows since each disjunction at the top contains a pair of dual atoms. $\square$

Lemma 7.19. *Let* α *be a valid formula in conjunctive normal form, with at least two conjuncts, such that each atom occurs as many times as its dual. Then there is a* *polynomial-size derivation of the following shape:*

$$\begin{array}{c} \bigwedge_i a_i \vee \bar{a}_i \\ \Big\Vert \mathsf{MS} \\ \alpha \end{array}$$

Proof. Since α is valid each of its disjunctions must contain a pair of dual atoms. If there are two such pairs in some disjunction then build the following derivation:

$$\mathsf{M}\,\dfrac{2{\cdot}\mathsf{S}\,\dfrac{[a \vee \bar{a}] \wedge [\beta \vee \gamma] \wedge \left[b \vee \bar{b} \vee \alpha\right]}{([a \vee \bar{a}] \wedge \beta) \vee \left(\gamma \wedge \left[b \vee \bar{b}\right]\right)}}{\left[a \vee \bar{a} \vee b \vee \bar{b} \vee \alpha\right] \wedge [\beta \vee \gamma]}$$

Read bottom-up, the number of pairs of dual atoms in the same disjunction has reduced and validity has been preserved, so we can repeatedly apply this construction until there are no disjunctions with two pairs of dual atoms.

Now each disjunction has exactly one pair of dual atoms, so match each other atom in a disjunction with an occurrence of its dual in another disjunction; the matching is bijective by the given condition.

We build the following derivation:

$$\mathsf{m}\,\dfrac{2{\cdot}\mathsf{s}\,\dfrac{\alpha \wedge \beta \wedge [a \vee \bar{a}]}{(\alpha \wedge \bar{a}) \vee (\beta \wedge a)}}{[\alpha \vee a] \wedge (\beta \wedge \bar{a})}$$

Read bottom-up, if a and $\bar{a}$ are a matching pair, the total number of matching pairs in distinct disjunctions has reduced and validity has been preserved, so we can repeatedly apply this construction to obtain a derivation of the required form. □

Proof of Thm. 7.16. Replacing each atom occurrence a_i in σ by $m_i \cdot a_i$ and $\bar{a}_i$ by $n_i \cdot \bar{a}_i$ ensures that there are equally many occurrences of a_i and $\bar{a}_i$. Now we can stitch together the constructions from Cor. 9.5 and Prop. 7.17 and Lemmata 7.18, 7.19, taking k to be the maximum of the values obtained for k in Prop. 7.17 and Lemma 7.18, to obtain a derivation of the required shape. □

Corollary 7.20. *Verifying the validity of a tautology τ can be reduced to determining the existence of a* MS*-rewrite path between two formulae in time polynomial in the size of the smallest Hilbert-Frege proof of τ.*

Proof. The premiss and conclusion of the derivations in Thm. 7.16 are governed by a single parameter, k. We simply run any algorithm that determines the existence of a MS-rewrite path between two formulae on the premiss and conclusion determined by each value of k, from 1 upwards, until it returns. $\square$

7.3.2 No polynomial-time characterisation for MS, conditionally

By the corollary above, any polynomial-time characterisation of MS would yield an algorithm verifying any tautology in time polynomial in the size of its smallest Hilbert-Frege proof. The existence of such an algorithm for a proof system, known as *weak automatisability*, was proved to be impossible for Hilbert-Frege systems in [BPR97], conditional on the assumption that integer factoring cannot be computed by polynomial-size circuits.

The definition we give below of weak automatisability differs slightly from that appearing in the literature, where usually an output proof is required in some fixed PPS, but it is straightforward to see that the two notions coincide.

Definition 7.21. A PPS P is *weakly automatisable* if there is a procedure verifying the validity of any tautology τ in time polynomial in the size of the smallest P-proof of τ.

Theorem 7.22 (Bonet et al.)**.** *If integer factoring is outside $P/poly$ then Hilbert-Frege systems are not weakly automatisable.*

Corollary 7.23. *If integer factoring is outside $P/poly$ then there is no polynomial-time characterisation of* MS.

Remark 7.24. Since it is exactly the logical information that is omitted in an atomic flow, it follows from the results in this section that atomic flows do not form a PPS, in the sense that they cannot be verified or sequentialised in polynomial-time, given an input premiss and conclusion, unless integer factoring is in $P/poly$. This conditionally refutes a conjecture of Guglielmi that atomic flows form an abstract proof system [GG08].

7.4 On linear inferences in general

In the previous sections we considered the specific rules s and m, but there are infinitely many other inferences one could consider. In addition to the proof theoretic interest of studying deep inference systems with different linear rules, or finding lower bounds for existing systems, there are also complexity theoretic motivations, due to the following result by Straßburger.

Proposition 7.25 (Straßburger)**.** *The set of all linear inferences is* coNP*-complete.*

This means that, rather than studying proof theory for the whole of propositional logic to make insights into computational complexity, we can restrict ourselves to the linear fragment. This arguably gives us a different viewpoint on the problem, since much of the work on read-once formulae and linear inferences are of a more combinatorial nature, e.g. [Gur77], [Str07].

MSU cannot derive every linear inference. This is immediate from Straßburger's result above, and since the length of paths can be assumed to be polynomial, under the assumption that $\mathsf{coNP} \neq \mathsf{NP}$. Unlike the case for TAUT, we know of no 'complete' set of linear inferences which can derive every other one, and designing such a set could be a first step towards understanding the structure of the set of linear inferences.

As a curiosity, we give below a linear inference on 10 variables that is not derivable in MS, improving on a 36 variable inference given by Straßburger [Str09], and conjecture that it is the minimal such inference.[4] By observing that there are no trivial atoms, we also have that it is independent of MSU.

Proposition 7.26. *The following is a linear inference that is not derivable in* MS*.*

$$\frac{[A \vee (B \wedge B')] \wedge [(C \wedge C') \vee (D \wedge D')] \wedge [(E \wedge E') \vee F]}{([C \vee E] \wedge [A \vee (C' \wedge E')]) \vee ([(B \wedge D) \vee F] \wedge [B' \vee D'])}$$

Proof. The inference is linear by inspection and its soundness can be checked mechanically. However we give an intuitive argument below, to give an idea of its meaning.

The inference is essentially an encoding of the pigeonhole principle with 3 pigeons and 2 holes. Consider the following grid:

A	$B \quad B'$
$C \quad C'$	$D \quad D'$
$E \quad E'$	F

The linear inference roughly[5] encodes the statement, "if each row contains a box whose variables are true then some column has two boxes with a true variable", which is clearly a tautology since there are more rows than columns. The use of multiple variables in some boxes is so that repetition of variables is avoided, ensuring linearity.

Using this interpretation, it is clear that any application of switch or medial leading to the conclusion must be from a formula not logically implied by the premiss. This can also be checked mechanically. $\square$

[4]Some progress on proving this via computational methods has been made by Šipraga in [Š12].

[5]Not exactly since not all combinations of variables in boxes are exhausted.

Corollary 7.27. *The above inference cannot be derived in* MSU*.*

Proof. If it could then some variable must be trivialised by Lemma 7.12, meaning we could substitute $\top$ for it in the premiss and $\bot$ in the conclusion and obtain a valid implication. Inspection shows that no variable has this property; the interpretation above makes it easier to verify this. $\square$

Similar inferences could be designed for all $n \times (n-1)$ grids, each one independent of all previous ones, and it is not difficult to see that adding these to KS would yield a system that has polynomial-size proofs of the pigeonhole principle tautologies considered in Chapt. 9, as noticed in [Str09]. However such a system would not be local since it would have rules of unbounded size. Nonetheless such systems would fall within the definition of a PPS, and would retain much of the theory already developed for deep inference, e.g. the manipulations of structural interactions in Chapt. 6 would still be valid.

Part IV

Complexity without cocontraction

By the quasipolynomial normalisation theorem of Chapt. 5 we have answered many of the questions surrounding the complexity of KS^+, or equivalently $\mathsf{KS} \cup \{\mathsf{ac}\uparrow\}$, however the task of classifying KS in the proof complexity hierarchy remains largely unaddressed.

Naïve attempts to generalise the normalisation theorem to KS fail, and the $\mathsf{ac}\uparrow$-elimination procedures induced by the results in Chapt. 6 can incur an exponential blowup; unsurprisingly many authors conjecture that there is a superpolynomial separation between the two systems, e.g. [BG09b] [Str09] [Das11], although this feeling is not unanimous in the deep inference community, e.g. [Jeř09] [Das12].

Nonetheless we show in this part that KS is a quite powerful proof system, compared to systems for which we have nontrivial lower bounds. In Chapt. 8 we give polynomial simulation of truth tables, and in Chapt. 9 we give short proofs of the propositional pigeonhole principle along with some variants and related principles.

Both chapters rely on the results of Chapt. 6. Sects. 9.2, 9.3 and 9.4, in particular, we consider a culmination of a lot of the ideas appearing this dissertation, and we hope exemplify the power of some of the techniques developed.

Chapter 8

A simple simulation in KS

In this chapter we give a polynomial simulation of truth tables in KS. We rely on the results of Chapt. 6, first designing a KS^+-simulation and then arguing that such proofs normalise to KS in polynomial time, by appealing to the structure of their atomic flows.

The normalisation procedure is quite simple, since there are boundedly many alternations between $\mathsf{ac}\downarrow$ and $\mathsf{ac}\uparrow$ nodes in the flows. The proofs in Chapt. 9 are more sophisticated, containing proofs of polylogarithmic length, but we use this chapter as an exercise to set up the arguments that occur later.

The material in this chapter is based on work that has appeared in [Das12] and also work that is in submission.

8.1 Truth tables and tableaux

KS polynomially simulates tree-like cut-free Gentzen sequent calculi since its rules are generalisations of Gentzen sequent rules. In the other direction Bruscoli and Guglielmi have proved in [BG09b] that the converse does not hold, via the so-called 'Statman' tautologies. We offer a new proof here, via a simulation of truth tables, appealing to the following result:

Proposition 8.1 (D'Agostino)**.** *The tree-like cut-free sequent calculus cannot polynomially simulate truth tables.*

Proof. See [D'A92]. □

To expand slightly on the above proposition, truth tables are efficient when there are exponentially many occurrences of each atom, and some such tautologies are hard for tree-like cut-free sequent calculi. One such example, used by D'Agostino, is simply the disjunction of every assignment on k propositional variables, what we call $\bigvee_{\mathcal{A}} \gamma_{\mathcal{A}}$

below. Such a formula has size $k \cdot 2^k$, although any tree-like cut-free sequent proof must contain at least $k!$ branches, while a truth table contains 2^k rows and $k \cdot 2^k$ columns.[1]

Lemma 8.2. KS^+ *polynomially simulates truth tables.*

Proof. Let τ be a tautology. For each partial assignment $\mathcal{A}$, defined on just those variables appearing in τ, and each formula A satisfied by $\mathcal{A}$ construct a derivation $\Phi_{\mathcal{A}}(A)$ by structural induction on A as follows:

$$\Phi_{\mathcal{A}}(a) \equiv a \quad , \quad \Phi_{\mathcal{A}}(B \wedge C) \equiv \Phi_{\mathcal{A}}(B) \wedge \Phi_{\mathcal{A}}(C) \quad , \quad \Phi_{\mathcal{A}}(B \vee C) \equiv {=}\frac{\Phi_{\mathcal{A}}(B)}{B \vee \mathsf{w}{\downarrow}\dfrac{\bot}{C}}$$

where, in the last case, when A is a disjunction, the disjunct B was chosen such that B is satisfied by $\mathcal{A}$. It is clear that each $\Phi_{\mathcal{A}}(\tau)$ has conclusion τ and premiss a conjunction of literals; moreover this conjunction of literals is satisfied by $\mathcal{A}$.

Let $\gamma_{\mathcal{A}}$ be the conjunction of all literals satisfied by $\mathcal{A}$, so that each variable appears exactly once. Then we can easily construct derivations $\begin{array}{c}\gamma_{\mathcal{A}} \\ \Big\| \{\mathsf{aw}{\uparrow},\mathsf{ac}{\uparrow}\} \\ \mathsf{pr}(\Phi_{\mathcal{A}}(\tau))\end{array}$.

Now construct a proof Ψ of $\bigvee_{\mathcal{A}} \gamma_{\mathcal{A}}$ in $\{\mathsf{ai}{\downarrow}, \mathsf{ac}{\uparrow}, \mathsf{s}, \mathsf{m}\}$ by induction on number of distinct variables, as shown below:

$$\textit{Base case}: \quad \mathsf{ai}{\downarrow}\frac{\top}{a \vee \bar{a}} \quad , \quad \textit{Inductive step}: \quad 2\cdot\mathsf{s}\frac{\begin{array}{c}\Psi \Big\| \{\mathsf{ai}{\downarrow},\mathsf{ac}{\uparrow},\mathsf{s},\mathsf{m}\} \\ \bigvee_{\mathcal{A}} \left({=}\dfrac{\gamma_{\mathcal{A}}}{\mathsf{ai}{\downarrow}\dfrac{\top}{a \vee \bar{a}} \wedge \mathsf{c}{\uparrow}\dfrac{\gamma_{\mathcal{A}}}{\gamma_{\mathcal{A}} \wedge \gamma_{\mathcal{A}}}} \right)\end{array}}{\bigvee_{\mathcal{A}} (a \wedge \gamma_{\mathcal{A}}) \vee \bigvee_{\mathcal{A}} (\bar{a} \wedge \gamma_{\mathcal{A}})}$$

[1] Notice that $k!$ grows quasipolynomially in 2^k, by taking logarithms and using the integral bound.

Finally we put these together and apply contractions to obtain a KS^+ proof of τ:

$$\begin{array}{c} \Psi \Big\Vert \{\mathsf{ai}\downarrow,\mathsf{ac}\uparrow,\mathsf{s},\mathsf{m}\} \\ \bigvee_{\mathcal{A}} \left[\begin{array}{c} \gamma_{\mathcal{A}} \\ \Big\Vert \{\mathsf{aw}\uparrow,\mathsf{ac}\uparrow\} \\ \mathsf{pr}(\Phi_{\mathcal{A}}(\tau)) \\ \Phi_{\mathcal{A}} \Big\Vert \{\mathsf{w}\downarrow\} \\ \tau \end{array}\right] \\ \mathsf{c}\downarrow \; \overline{\overline{\hphantom{xxxxxxxxxxxxxxxx}}} \\ \tau \end{array}$$

It is clear that the derivations inside the large square brackets have size polynomial in $|\tau|$, which is the number of columns in a truth table, and the number of these derivations appearing in disjunction is just the number of assignments, which is the number of rows in a truth table. □

Theorem 8.3. KS *polynomially simulates truth tables.*

Proof. Notice that, in the above simulation, all $\mathsf{ac}\uparrow$ steps are above all $\mathsf{ac}\downarrow$ steps, and so the associated flows will have bounded length. The result follows by Thm. 6.22 and Prop. 6.24. □

Corollary 8.4. *Tree-like cut-free sequent calculi cannot polynomially simulate* KS.

Proof. Immediate from Prop. 8.1 and Thm. 8.3. □

Note that D'Agostino's separation is only quasipolynomial, and so our separation is also only quasipolynomial, while the proof using the Statman tautologies in [BG09b] yields an exponential separation. Nonetheless, an exponential separation follows from the proofs of variants of the pigeonhole principle in Chapt. 9.

8.2 Some remarks

We point out that, in an earlier version of this dissertation, we attempted to use the result of Sect. 6.5 to establish KS-simulations of fragments of sequent and Resolution systems, appealing to the loop structure of associated flows of KS^+-simulations rather than their length. Unfortunately the analysis of these translations were incomplete and proved problematic, and so we omit them from the final version of this dissertation.

Chapter 9

Some combinatorial principles in deep inference

In this chapter we consider propositional encodings of certain combinatorial principles and examine the size of their proofs in KS. In particular we are interested in the pigeonhole principle, that if there are n pigeons sitting in $n-1$ holes then there must be some hole with two pigeons sitting in it.

This can be encoded in propositional logic as follows,

$$\bigwedge_{i=1}^{n} \bigvee_{j=1}^{n-1} a_{ij} \rightarrow \bigvee_{j=1}^{n-1} \bigvee_{i=1}^{n-1} \bigvee_{i'=i+1}^{n} a_{ij} \wedge a_{i'j}$$

where a_{ij} should be interpreted as "pigeon i sits in hole j".

These tautologies have been used to obtain exponential lower bounds for various proof systems, including cut-free sequent systems, Resolution, and bounded-depth Hilbert-Frege systems. They are considered somewhat of a bench mark in proof complexity [Raz02].

By giving small proofs of the propositional pigeonhole principle and some variants in KS we thus obtain exponential separations from all these systems, as well as demonstrating the power of some of the tools we have developed, and in turn the proof complexity theoretic strength of KS.

The material in Sect. 9.1 of this chapter is based on work that has appeared in [Das12] and work that is in submission, and the results of later sections is based on work that has appeared in [Das14].

9.1 Variants of the pigeonhole principle

Notice that the encoding of the pigeonhole principle we gave above allows the mapping from pigeons to holes to be many-many, i.e. an arbitrary total relation. We consider here weaker variants that place restrictions on the nature of this relation. The *functional* variant insists that the mapping is many-one, i.e. a function, while the *onto* variant insists that each hole is occupied. Intuitively, these two variants are 'weaker' than the unrestricted version, and the variant containing both criteria, the *onto functional* pigeonhole principle is weaker still. We will see this more clearly in the following definition.

Definition 9.1 (Pigeonhole principles)**.** We define the following formulae,

$$\mathsf{PHP}_n \equiv \bigvee_{i=1}^{n} \bigwedge_{j=1}^{n-1} \bar{a}_{ij} \vee \bigvee_{i=1}^{n} \bigvee_{j<j'} (a_{ij} \wedge a_{ij'}), \quad \mathsf{F}_n \equiv \bigvee_{j=1}^{n-1} \bigvee_{i<i'} (a_{ij} \wedge a_{i'j}), \quad \mathsf{O}_n \equiv \bigvee_{j=1}^{n-1} \bigwedge_{i=1}^{n} \bar{a}_{ij}$$

and denote by FPHP_n, OPHP_n and OFPHP_n the formulae obtained by putting in disjunction the associated formulae, i.e. $\mathsf{FPHP}_n \equiv \mathsf{F}_n \vee \mathsf{PHP}_n$, $\mathsf{OPHP}_n \equiv \mathsf{O}_n \vee \mathsf{PHP}_n$ and $\mathsf{OFPHP}_n \equiv \mathsf{O}_n \vee \mathsf{F}_n \vee \mathsf{PHP}_n$.

We can see in the above definition that any variant can be obtained from a stronger variant, i.e. one with a subset of disjuncts, by a simple application of generic weakening $\mathsf{w}\downarrow$. Consequently upper bounds on the size of proofs of one variant yield upper bounds for all weaker variants, and lower bounds vice-versa.

The following result was proved independently by Beame et al. and Krajíček et al.

Theorem 9.2. *Bounded-depth Hilbert-Frege systems have only exponential-size proofs of* OFPHP_n.

Proof. See [PBI93] and [KPW95]. □

Corollary 9.3. *Cut-free sequent calculi, Resolution and bounded-depth Hilbert-Frege systems have only exponential-size proofs of all variants of the pigeonhole principle.*

Proof. All the systems are just special cases of bounded-depth Hilbert-Frege systems, and a proof of any variant can be simply extended to one of OFPHP_n. □

Theorem 9.4 (Buss)**.** *There are polynomial-size Hilbert-Frege proofs of* PHP_n*, and so all variants of the pigeonhole principle.*

Proof. See [Bus87]. □

Recall Prop. 5.7, that any SKS-proof Φ of a tautology τ over atoms $a_1, \dots, a_n$ can be polynomially transformed into an $\mathsf{SKS} \setminus \{\mathsf{ai}\downarrow, \mathsf{ai}\uparrow\}$-derivation Φ' from $\bigwedge\limits_{i=1}^{n} [a_i \vee \bar{a}_i]$ to $\tau \vee \bigvee\limits_{i=1}^{n} (a_i \wedge \bar{a}_i)$. We give the following consequence of this result, which will prove useful in the constructions that follow.

Corollary 9.5 (of Prop. 5.7)**.** *Let τ be a tautology over the atoms $a_1, \dots, a_n$. Every SKS-proof Φ of τ can be polynomially transformed to a KS proof of $\tau \vee \bigvee_i (a_i \wedge \bar{a}_i)$.*

Proof. Let Φ' be the derivation obtained from Φ by Prop. 5.7, and append to the top of Φ' the following derivation,

$$= \frac{\top}{\bigwedge\limits_{i=1}^{n} \mathsf{ai}\downarrow \dfrac{\top}{a_i \vee \bar{a}_i}}$$

so that we have an KS^+-proof Φ'' of $\tau \vee \bigvee_i (a_i \wedge \bar{a}_i)$. Now notice that, in Φ'', every $\mathsf{ac}\uparrow$-step is above every $\mathsf{ac}\downarrow$-step and so its flow will have bounded length. By Thm. 6.22 and Prop. 6.24 it then follows that this proof can be polynomially transformed to a KS-proof of the required form. □

The significance of this result is that, if we know there is an SKS-proof of a tautology, then we can transform it into a KS-proof of that tautology in disjunction with some trivial contradictions. If we can find derivations from each of these contradictions to the tautology we want to prove, then we can put them in disjunction and apply contraction to build a proof of the tautology.

This idea was noticed by Jeřábek in [Jeř09], where he proved the following result.

Lemma 9.6 (Jeřábek)**.** *There are polynomial-size KS^+-proofs of FPHP_n and OPHP_n.*

Proof. By Thm. 9.4 and Cor. 9.5, and since SKS is polynomially equivalent to Hilbert-Frege systems, there are polynomial-size KS-proofs of $\mathsf{PHP}_n \vee \bigvee_{i,j} (a_{ij} \wedge \bar{a}_{ij})$. For each atom a_{st} we construct a derivations $\Phi_n^{a_{st}}$ in $\mathsf{KS}^+ \setminus \{\mathsf{ac}\downarrow\}$ with premiss $a_{st} \wedge \bar{a}_{st}$ and conclusion FPHP_n respectively as shown below on the left. We then put these together and apply contractions to obtain proofs of FPHP_n, as shown on the right.

$$
\mathsf{w}{\downarrow}\,\dfrac{\mathsf{2\cdot s}\,\dfrac{=\,\dfrac{a_{st}\wedge\bar{a}_{st}}{a_{st}\wedge\bar{a}_{st}\wedge\,\mathsf{i}{\downarrow}\,\dfrac{\top}{\bigwedge_{j\neq t}\bar{a}_{sj}\vee\bigvee_{j\neq t}a_{sj}}}}{\bigwedge_{j}\bar{a}_{sj}\vee\left(\begin{array}{c}(n-2)\cdot\mathsf{ac}{\uparrow}\,\dfrac{a_{st}}{a_{st}\wedge\cdots a_{st}}\wedge\bigvee_{j\neq t}a_{sj}\\ \Big\Vert\{\mathsf{s}\}\\ \bigvee_{j\neq t}a_{st}\wedge a_{sj}\end{array}\right)}}{\mathsf{FPHP}_n}
\quad , \quad
\begin{array}{c}
\Big\Vert\mathsf{KS}\\
\left[\mathsf{PHP}_n\vee\bigvee_{i,j}\left(\begin{array}{c}a_{ij}\wedge\bar{a}_{ij}\\ \Phi_n^{a_{ij}}\Big\Vert\mathsf{KS}^{+}\setminus\{\mathsf{ac}{\downarrow}\}\\ \mathsf{FPHP}_n\end{array}\right)\right]\\
\Big\Vert\{\mathsf{c}{\downarrow}\}\\
\mathsf{FPHP}_n
\end{array}
$$

We can construct similar derivations from $a_{st}\wedge\bar{a}_{st}$ to OPHP_n, given below, and put them together in the same way to obtain proofs of OPHP_n in KS^{+}.

$$
\mathsf{w}{\downarrow}\,\dfrac{\mathsf{2\cdot s}\,\dfrac{=\,\dfrac{a_{st}\wedge\bar{a}_{st}}{a_{st}\wedge\bar{a}_{st}\wedge\,\mathsf{i}{\downarrow}\,\dfrac{\top}{\bigwedge_{i\neq s}\bar{a}_{it}\vee\bigvee_{i\neq s}a_{it}}}}{\bigwedge_{i}\bar{a}_{it}\vee\left(\begin{array}{c}(n-2)\cdot\mathsf{ac}{\uparrow}\,\dfrac{a_{st}}{a_{st}\wedge\cdots\wedge a_{st}}\wedge\bigvee_{i\neq s}a_{it}\\ \Big\Vert\{\mathsf{s}\}\\ \bigvee_{i\neq s}a_{st}\wedge a_{it}\end{array}\right)}}{\mathsf{OPHP}_n}
$$

□

It turns out that these proofs can be transformed to KS-proofs in polynomial time by the results of Chapt. 6.

Theorem 9.7. *There are polynomial-size proofs in* KS *of* FPHP_n*,* OPHP_n *and so also* OFPHP_n*.*

Proof. In the proofs of FPHP_n constructed above, Lemma 9.6, notice that the only $\mathsf{ac}{\uparrow}$-steps occur in $\Phi_n^{a_{st}}$ where there are also no $\mathsf{ac}{\downarrow}$-steps, and similarly for OPHP_n. It follows that there are only two alternations between $\mathsf{ac}{\downarrow}$ and $\mathsf{ac}{\uparrow}$ steps in the path of any atom from an $\mathsf{ai}{\downarrow}$-step, and so the atomic flows of these proofs will have bounded length. The result follows by Thm. 6.22 and Prop. 6.24. □

Corollary 9.8. *Cut-free sequent calculi, Resolution and bounded-depth Hilbert-Frege systems are exponentially separated from* KS*.*

Proof. Immediate from Cor. 9.3 and Thm. 9.7. □

9.2 The unrestricted pigeonhole principle

We now turn our attention to the unrestricted variant of the pigeonhole principle, when the mapping from pigeons to holes is allowed to be many-many. In many systems, e.g. bounded-depth Hilbert-Frege, there is not much difference between the variants since they are interderivable. However for monotone-like systems they seem to differ greatly, and the methods of the previous section do not seem to generalise, as has been noted by previous authors [Jeř09] [AGP02].

Rather than generalising previous results, this section is inspired by the work of Atserias et al. in [AGG00], where monotone threshold formulae are employed to carry out basic counting arguments. We use the same proof idea although our main construction, proofs that permute arguments of threshold formulae, are somewhat more involved than theirs. Their approach provides proofs corresponding to transpositions and then appeals to the fact that every permutation can be written as a composition of transpositions. However formalising that construction in KS^+ results in proofs whose atomic flows have polynomial length, and so normalise to KS-proofs of exponential size.

Instead we notice that the specific permutation required, corresponding to the transposition of a matrix, can be expressed simply as a composition of *interleavings* and formalising this decomposition results in proofs whose flows have polylogarithmic length, and so normalise in quasipolynomial time.

9.2.1 Monotone and normal derivations

Recall that a monotone derivation is just an $\mathsf{SKS} \setminus \{\mathsf{ai}\downarrow, \mathsf{ai}\uparrow\}$-derivation.

Definition 9.9. A monotone derivation is said to be *normal* if it has the following shape:

$$\begin{array}{c} A \\ \Vert \{\mathsf{aw}\uparrow,\mathsf{ac}\uparrow,\mathsf{s},\mathsf{m}\} \\ B \\ \Vert \{\mathsf{aw}\downarrow,\mathsf{ac}\downarrow,\mathsf{s},\mathsf{m}\} \\ C \end{array}$$

Proposition 9.10. *A monotone derivation* Φ *whose flow is in normal form for* norm *can be polynomially transformed into a normal derivation with same premiss and conclusion.*

Proof. We have that the $\{\mathsf{ac}\downarrow, \mathsf{ac}\uparrow\}$-nodes, the $\mathsf{aw}\downarrow$-nodes and the $\mathsf{aw}\uparrow$-nodes form disconnected components of $fl(\Phi)$ and that all $\mathsf{ac}\uparrow$-nodes are above all $\mathsf{ac}\downarrow$-nodes, by a similar argument to that for Prop. 6.15. Notice also that $\mathsf{aw}\uparrow$ and $\mathsf{ac}\uparrow$ permute above logical steps, and dually $\mathsf{aw}\downarrow$ and $\mathsf{ac}\downarrow$ permute below logical steps, whence the result follows. □

Like in Sect. 5.1, many of our constructions are inductions, so we use the following result to omit tedious base cases.

Corollary 9.11. *Normal derivations are monotone implicationally complete.*

Proof. Monotone derivations are implicationally complete by Prop. 5.5, and can be transformed into a derivation whose flow is in normal form for norm by Cor. 6.9, whence the result follows by Prop. 9.10 above. □

The significance of normal derivations is that they can be efficiently transformed into KS-proofs of the implication they derive, as demonstrated in the following proposition.

Proposition 9.12. *A normal derivation* $\begin{array}{c} A \\ \Phi \| \{\mathsf{aw}\uparrow,\mathsf{ac}\uparrow,\mathsf{s},\mathsf{m}\} \\ B \\ \Psi \| \{\mathsf{aw}\downarrow,\mathsf{ac}\downarrow,\mathsf{s},\mathsf{m}\} \\ C \end{array}$ *can be transformed in linear time to a* KS*-proof of* $\bar{A} \vee C$.

Proof. Let $\begin{array}{c} \bar{B} \\ \bar{\Phi} \| \{\mathsf{aw}\downarrow,\mathsf{ac}\downarrow,\mathsf{s},\mathsf{m}\} \\ \bar{A} \end{array}$ be the dual of Φ, as defined in Dfn. 3.9. Now construct the required derivation as follows: $\mathsf{i}\downarrow \dfrac{\top}{\begin{array}{ccc} \bar{B} & & B \\ \bar{\Phi}\| & \vee & \Psi\| \\ \bar{A} & & C \end{array}}$. □

9.2.2 Interleavings and transposition of threshold inputs

Recall the definition of threshold formulae th^n_k from Dfn. 5.2. It will be convenient in this section to sometimes represent the inputs of these formulae as a matrix of variables, denoted by bold uppercase letters $\boldsymbol{A}, \boldsymbol{B}$, etc. We associate with such a matrix the vector obtained by reading it rows-first. In this way the transpose of a matrix is equivalent to the vector obtained by reading it columns-first.

Throughout this and later subsections the variables m and n are powers of 2 and $m \leq n$. All proofs are monotone unless otherwise mentioned.

Definition 9.13 (Interleaving)**.** For $\boldsymbol{a} = (a_1, \ldots, a_n), \boldsymbol{b} = (b_1, \ldots, b_n)$ let $\boldsymbol{a} \mathbin{|||} \boldsymbol{b}$ denote the *interleaving* of $\boldsymbol{a}$ with $\boldsymbol{b}$: $(a_1, b_1, \ldots, a_n, b_n)$.

More generally, we denote by $\boldsymbol{a} \mathbin{|||}_m \boldsymbol{b}$ the m-interleaving of $\boldsymbol{a}$ with $\boldsymbol{b}$:

$$(a_1, \ldots, a_m, b_1, \ldots b_m, \cdots, a_{n-m+1}, \ldots, a_n, b_{n-m+1}, \ldots, b_n)$$

Recall the distributivity laws from Dfn. 4.4; we will write $\mathsf{dist}\uparrow = \{\mathsf{d}_2\uparrow, \mathsf{d}_1\uparrow\}$ and $\mathsf{dist}\downarrow = \{\mathsf{d}_2\downarrow, \mathsf{d}_1\downarrow\}$ and use these as abbreviations for the derivations of those laws. In particular notice that $\mathsf{dist}\uparrow$-derivations are free of $\mathsf{ac}\downarrow$-steps and $\mathsf{dist}\downarrow$-derivations are free of $\mathsf{ac}\uparrow$-steps; we will use this property when calculating the length of atomic flows.

Lemma 9.14. *There are monotone derivations,*

$$\begin{array}{c} \mathsf{th}^{2n}_r(\boldsymbol{a}, \boldsymbol{b}) \\ \Big\Vert \\ \mathsf{th}^{2n}_r(\boldsymbol{a} \mathbin{|||}_m \boldsymbol{b}) \end{array}$$

whose flows have length $O(\log n - \log m)$ and width $O(r)$.

Proof. We use the following identity,

$$(\boldsymbol{a}, \boldsymbol{b}) \mathbin{|||}_m (\boldsymbol{c}, \boldsymbol{d}) = (\boldsymbol{a} \mathbin{|||}_m \boldsymbol{c}, \boldsymbol{b} \mathbin{|||}_m \boldsymbol{d})$$

to give an inductive step from n to $2n$ below,

$$\begin{array}{lc}
 & \mathsf{th}^{4n}_r(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}) \\ \hline
= & \displaystyle\bigvee_{s+t=r} \left(\begin{array}{c} = \dfrac{\mathsf{th}^{2n}_s(\boldsymbol{a}, \boldsymbol{b})}{\displaystyle\bigvee_{i+j=s} \mathsf{th}^{n}_i(\boldsymbol{a}) \wedge \mathsf{th}^{n}_j(\boldsymbol{b})} \wedge = \dfrac{\mathsf{th}^{2n}_t(\boldsymbol{c}, \boldsymbol{d})}{\displaystyle\bigvee_{k+l=t} \mathsf{th}^{n}_k(\boldsymbol{c}) \wedge \mathsf{th}^{n}_l(\boldsymbol{d})} \\ \Big\Vert {\scriptstyle \mathsf{dist}\uparrow} \\ \displaystyle\bigvee_{\substack{i+j=s \\ k+l=t}} \big(\mathsf{th}^{n}_i(\boldsymbol{a}) \wedge \mathsf{th}^{n}_j(\boldsymbol{b})\big) \wedge (\mathsf{th}^{n}_k(\boldsymbol{c}) \wedge \mathsf{th}^{n}_l(\boldsymbol{d})) \end{array} \right) \\ \hline
= & \displaystyle\bigvee_{s'+t'=r} \left(\begin{array}{c} \displaystyle\bigvee_{\substack{i+k=s' \\ j+l=t'}} (\mathsf{th}^{n}_i(\boldsymbol{a}) \wedge \mathsf{th}^{n}_k(\boldsymbol{c})) \wedge (\mathsf{th}^{n}_j(\boldsymbol{b}) \wedge \mathsf{th}^{n}_l(\boldsymbol{d})) \\ \Big\Vert {\scriptstyle \mathsf{dist}\downarrow} \\ \begin{array}{ccc} \mathsf{th}^{2n}_{s'}(\boldsymbol{a}, \boldsymbol{c}) & & \mathsf{th}^{2n}_{t'}(\boldsymbol{b}, \boldsymbol{d}) \\ {\scriptstyle IH}\Big\Vert & \wedge & {\scriptstyle IH}\Big\Vert \\ \mathsf{th}^{2n}_{s'}(\boldsymbol{a} \mathbin{|||}_m \boldsymbol{c}) & & \mathsf{th}^{2n}_{t'}(\boldsymbol{b} \mathbin{|||}_m \boldsymbol{d}) \end{array} \end{array} \right) \\ \hline
= & \mathsf{th}^{4n}_r((\boldsymbol{a}, \boldsymbol{b}) \mathbin{|||}_m (\boldsymbol{c}, \boldsymbol{d}))
\end{array}$$

where derivations marked IH are obtained by the inductive hypothesis.

Each inductive step adds two alternations of $\mathsf{ac}\uparrow$ and $\mathsf{ac}\downarrow$ nodes to a flow, in the form of the distributivity steps, on top of two copies of the inductive hypothesis in parallel. The $\mathsf{dist}\uparrow$ steps duplicate each atom at most r times, whence the bound on width is obtained, and the induction terminates in $O(\log \frac{n}{m})$ steps, whence the bound on length is obtained. $\square$

Observation 9.15. *For matrices $\boldsymbol{B}$ and $\boldsymbol{C}$ of equal dimensions we have:*[1]

$$\begin{pmatrix}\boldsymbol{A} & \boldsymbol{B}\\ \boldsymbol{C} & \boldsymbol{D}\end{pmatrix}^{\intercal} = \begin{pmatrix}\boldsymbol{A}^{\intercal} & \boldsymbol{C}^{\intercal}\\ \boldsymbol{B}^{\intercal} & \boldsymbol{D}^{\intercal}\end{pmatrix}$$

Recall that a matrix of variables is equivalent to the vector obtained from a rows-first reading of it.

Theorem 9.16 (Transposition)**.** *There are monotone derivations,*

$$\begin{array}{c}\mathsf{th}^n_k(\boldsymbol{X})\\ \Big\| \\ \mathsf{th}^n_k(\boldsymbol{X}^{\intercal})\end{array}$$

whose flows have length $O(\log^2 n)$ and width $O(k)$.

Proof. Let $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}$ be the four quadrants of $\boldsymbol{X}$. We give the inductive step below,

$$\begin{array}{rc}
 & \mathsf{th}^{2n}_k\begin{pmatrix}\boldsymbol{A} & \boldsymbol{B}\\ \boldsymbol{C} & \boldsymbol{D}\end{pmatrix} \\ \cline{2-2}
= & \displaystyle\bigvee_{i+j=k}\left(\begin{array}{ccc}
\mathsf{th}^n_i\begin{pmatrix}\boldsymbol{A} & \boldsymbol{B}\end{pmatrix} & & \mathsf{th}^n_i\begin{pmatrix}\boldsymbol{C} & \boldsymbol{D}\end{pmatrix}\\
{}^{IH}\Big\| & \wedge & {}^{IH}\Big\| \\
\mathsf{th}^n_i\begin{pmatrix}\boldsymbol{A}^{\intercal}\\ \boldsymbol{B}^{\intercal}\end{pmatrix} & & \mathsf{th}^n_i\begin{pmatrix}\boldsymbol{C}^{\intercal}\\ \boldsymbol{D}^{\intercal}\end{pmatrix}
\end{array}\right) \\ \cline{2-2}
= & \mathsf{th}^{2n}_k\left(\begin{pmatrix}\boldsymbol{A}^{\intercal}\\ \boldsymbol{B}^{\intercal}\end{pmatrix}, \begin{pmatrix}\boldsymbol{C}^{\intercal}\\ \boldsymbol{D}^{\intercal}\end{pmatrix}\right) \\
 & {}^{\text{interleave}}\Big\| \\
 & \mathsf{th}^{2n}_k\begin{pmatrix}\boldsymbol{A}^{\intercal} & \boldsymbol{C}^{\intercal}\\ \boldsymbol{B}^{\intercal} & \boldsymbol{D}^{\intercal}\end{pmatrix}
\end{array}$$

where the derivations marked IH are obtained by the inductive hypothesis and Obs. 9.15,

[1]Of course, in any such situation, $\boldsymbol{A}$ and $\boldsymbol{D}$ will also have equal dimensions.

and the derivation marked 'interleave' is obtained by applying Lemma 9.14 to interleave the rows of the two matrices.

Each inductive step adds an interleaving below two copies of the inductive hypothesis in parallel, thereby adding $O(\log n)$ to the length of the associated flow and maintaining its width of $O(k)$, by Lemma 9.14. The induction terminates in $O(\log n)$ steps, whence the upper bound on length is obtained. $\square$

9.2.3 From threshold formulae to the pigeonhole principle

The previous section showed that there are 'short' derivations transposing a matrix of the arguments of a threshold formula. We show here how such derivations are used to obtain quasipolynomial-size proofs of the propositional pigeonhole principle.

Definition 9.17. We define the following:

$$\mathsf{LPHP}_n \equiv \bigwedge_{i=1}^{n} \bigvee_{j=1}^{n-1} a_{ij} \quad , \quad \mathsf{RPHP}_n \equiv \bigvee_{j=1}^{n-1} \bigvee_{i=1}^{n} \bigvee_{i'=i+1}^{n} (a_{i'j} \wedge a_{ij})$$

Notice that PHP_n, as defined before, is just $\overline{\mathsf{LPHP}_n} \vee \mathsf{RPHP}_n$.

Definition 9.18. Let $\perp_{mn}$ be the $(m \times n)$ matrix with the constant $\perp$ at every entry. Define $\boldsymbol{P}_n = \begin{pmatrix} (a_{ij}) & \perp_{n1} \end{pmatrix}$, with i, j ranging as in Dfn. 9.17. I.e. $\boldsymbol{P}_n$ is obtained by extending (a_{ij}) with an extra column of $\perp$-entries, so that it is a square matrix.

Our goal is to prove the following result, from which we can extract proofs of PHP_n in KS by earlier results.

Proposition 9.19. *There are polynomial-size monotone derivations,*

$$\begin{array}{c} \mathsf{LPHP}_n \\ \Vert \\ \mathsf{th}_n^{n^2}(\boldsymbol{P}_n) \end{array} \quad , \quad \begin{array}{c} \mathsf{th}_n^{n^2}(\boldsymbol{P}_n^{\mathsf{T}}) \\ \Vert \\ \mathsf{RPHP}_n \end{array}$$

of bounded length.

Before we can give a proof, we need some intermediate results. It should be pointed out that similar results were provided in [AGG00] for the monotone sequent calculus, which could be translated to deep inference and normalised, but we include them for completeness. These intermediate results are fairly routine, and there is nothing intricate from the point of view of complexity.

Proposition 9.20. *There are polynomial-size normal derivations*

$$\begin{array}{c} \mathsf{th}^n_l(\boldsymbol{a}) \\ \Big\Vert \\ \mathsf{th}^n_k(\boldsymbol{a}) \end{array}$$

for $l \geq k$.

Proof. We give an inductive step from n to $2n$,

$$\begin{array}{c} = \dfrac{\mathsf{th}^{2n}_l(\boldsymbol{a},\boldsymbol{b})}{\displaystyle\bigvee_{i+j=l}\left(\begin{array}{ccc} \mathsf{th}^n_i(\boldsymbol{a}) & & \mathsf{th}^n_j(\boldsymbol{b}) \\ {\scriptstyle IH}\Big\Vert & \wedge & {\scriptstyle IH}\Big\Vert \\ \mathsf{th}^n_{i'}(\boldsymbol{a}) & & \mathsf{th}^n_{j'}(\boldsymbol{b}) \end{array}\right)} \\ \Big\Vert{\scriptstyle \{\mathsf{w}\downarrow,\mathsf{c}\downarrow\}} \\ \mathsf{th}^{2n}_k(\boldsymbol{a},\boldsymbol{b}) \end{array}$$

where i' and j' are chosen such that $i' \leq i$, $j' \leq j$ and $i' + j' = k$, and derivations marked IH are obtained by the inductive hypothesis. $\square$

Lemma 9.21. *There are polynomial-size normal derivations:*

$$\begin{array}{c} \mathsf{th}^{2n}_{r+s}(\boldsymbol{a},\boldsymbol{b}) \\ \Big\Vert \\ \mathsf{th}^n_{r+1}(\boldsymbol{a}) \vee \mathsf{th}^n_s(\boldsymbol{b}) \end{array}$$

Proof. Notice that if $i + j = r + s$ then $i > r$ or $j \geq s$. We give a construction below,

$$= \dfrac{= \dfrac{\mathsf{th}^{2n}_{r+s}(\boldsymbol{a},\boldsymbol{b})}{\displaystyle\bigvee_{i+j=r+s} \mathsf{th}^n_i(\boldsymbol{a}) \wedge \mathsf{th}^n_j(\boldsymbol{b})}}{{\scriptstyle s\cdot\mathsf{c}\downarrow}\dfrac{\displaystyle\bigvee_{\substack{i>r \\ j=r+s-i}}\left(\begin{array}{c} = \dfrac{\mathsf{th}^n_i(\boldsymbol{a}) \wedge {\scriptstyle \mathsf{w}\uparrow}\dfrac{\mathsf{th}^n_j(\boldsymbol{b})}{\top}}{\mathsf{th}^n_i(\boldsymbol{a})} \\ {\scriptstyle \Phi}\Big\Vert \\ \mathsf{th}^n_{r+1}(\boldsymbol{a}) \end{array}\right)}{\mathsf{th}^n_{r+1}(\boldsymbol{a})} \vee {\scriptstyle r\cdot\mathsf{c}\downarrow}\dfrac{\displaystyle\bigvee_{\substack{j\geq s \\ i=r+s-j}}\left(\begin{array}{c} = \dfrac{{\scriptstyle \mathsf{w}\uparrow}\dfrac{\mathsf{th}^n_i(\boldsymbol{a})}{\top} \wedge \mathsf{th}^n_j(\boldsymbol{b})}{\mathsf{th}^n_j(\boldsymbol{b})} \\ {\scriptstyle \Psi}\Big\Vert \\ \mathsf{th}^n_s(\boldsymbol{b}) \end{array}\right)}{\mathsf{th}^n_s(\boldsymbol{b})}}$$

where Φ and Ψ denote derivations obtained by Prop. 9.20. $\square$

Lemma 9.22. *Let $\boldsymbol{a}^1, \dots, \boldsymbol{a}^m$ be vectors of distinct variables. Then there are polynomial-size normal derivations:*

$$\begin{array}{c} \bigvee\limits_{r=1}^{m} \mathsf{th}_k^n(\boldsymbol{a}^r) \\ \| \\ \mathsf{th}_k^{mn}(\boldsymbol{a}^r)_{r=1}^n \end{array} \quad , \quad \begin{array}{c} \bigwedge\limits_{r=1}^{m} \mathsf{th}_k^n(\boldsymbol{a}^r) \\ \| \\ \mathsf{th}_{mk}^{mn}(\boldsymbol{a}^r)_{r=1}^n \end{array}$$

Proof. Simply apply $=$ and $\mathsf{w}\!\downarrow$ to fill out the formula. □

We are now in a position to prove Prop. 9.19.

Proof of Prop. 9.19. Recursively apply Lemma 9.21 to $\mathsf{th}_n^{n^2}(\boldsymbol{P}_n^\intercal)$, always setting $r = s$ or $r = s+1$, until a disjunction of threshold formulae with n inputs each is obtained. By the ordering of the literals in $\boldsymbol{P}_n^\intercal$ these threshold formulae will have as inputs $(a_{ij})_{i=1}^n$ for some j, or all $\bot$; in the latter case any such formula is equivalent to $\bot$, since the threshold will be at least 1, and so can be ignored.

In the former case, by the choice of r and s at each stage, we have that the threshold of each of these formulae is at least 2. Now we can apply Prop. 9.20 so that all thresholds are exactly 2, whence RPHP_n can be easily derived.

For the other derivation, construe each literal a_{ij} as a threshold formula $\mathsf{th}_1^1(a_{ij})$ and apply Lemma 9.22 to obtain a derivation from LPHP_n to $\mathsf{th}_n^{n^2}(\boldsymbol{P})$. □

Theorem 9.23. *There are normal derivations,*

$$\begin{array}{c} \mathsf{LPHP}_n \\ \| \\ \mathsf{RPHP}_n \end{array}$$

of size $n^{O(\log^2 n)}$.

Proof. By Thm. 9.16 and Prop. 9.19 there are monotone derivations from LPHP_n to RPHP_n whose flows have length $O(\log^2 n)$ and width $O(n)$. The result then follows by Thm. 6.22 and Prop. 6.24. □

Corollary 9.24. *There are KS-proofs of PHP_n of size $n^{O(\log^2 n)}$.*

Proof. Immediate from Thm. 9.23 and Prop. 9.12. □

9.2.4 The case when n is not a power of 2

Though we have assumed that n is a power of 2 throughout this section, the proof is actually sufficient for all n, as pointed out in [AGG00].

For $r \leq s$ given, define $\mathsf{LPHP}_s(r)$ by substituting $\bot$ for every atom a_{ij} where $i > r$ or $j \geq r$, and define $\mathsf{RPHP}_s(r)$ similarly.

Observation 9.25. *There are derivations* $\begin{array}{c}\mathsf{LPHP}_s(r)\\ \|\varnothing\\ \mathsf{LPHP}_r\end{array}$ *and* $\begin{array}{c}\mathsf{RPHP}_r\\ \|\{\mathsf{aw}\uparrow\}\\ \mathsf{RPHP}_s(r)\end{array}$.

So to construct proofs of PHP_r when r is not a power of two we simply construct the proof for the next power of 2 under the above substitutions and simplify units in the premiss and conclusion by the above observation.

9.3 Arbitrary permutations of threshold inputs

Interleavings by themselves do not form a generating set for the symmetric group, and so cannot be used to generate derivations for arbitrary permutations of inputs of threshold formulae. However a generalisation of them, corresponding to the set of riffle shuffles on a deck of cards, do form such a set. In this section we show how they may be used to generate arbitrary permutations on the inputs of threshold formulae. Essentially we are using merge-sort at the meta-level to control the construction of our deep inference proofs.

Throughout this section we assume that all trees are binary, and omit routine analysis of the complexity of proofs.

Recall that our original definition of threshold formulae used a symmetric divide-and-conquer strategy, generated from a complete binary tree in the natural way. In this section it will be useful to have a more general definition of threshold formulae, based on any tree decomposition of its arguments.

Definition 9.26. For a tree T, let $d(T)$ denote its depth, $l(T)$ its number of leaves and $|T|$ denote its number of nodes. For a binary tree T, let T_0 denote its left subtree (from the root) and T_1 its right. Thus any string $\sigma \in \{0,1\}^k$ determines an unique subtree T_σ of T, for $k \leq d(T)$.

Definition 9.27 (General threshold formulae)**.** For a binary tree T and vectors $\boldsymbol{a}, \boldsymbol{b}$ with $|\boldsymbol{a}| = l(T_0), |\boldsymbol{b}| = l(T_1)$, define

$$\mathsf{th}_k^T(\boldsymbol{a}, \boldsymbol{b}) \equiv \bigvee_{i+j=k} \mathsf{th}_i^{T_0}(\boldsymbol{a}) \wedge \mathsf{th}_j^{T_1}(\boldsymbol{b})$$

with the base cases the same as in Dfn. 5.2.

Observation 9.28. *For a tree T, $|\mathsf{th}_k^T(\boldsymbol{a})| = l(T)^{O(d(T))}$.*

What we define as a shuffle below corresponds to the common riffle method of shuffling a deck of cards: cut the deck anywhere, partitioning it into a left and right part, and then interleave these in any way, maintaining the relative order of cards from both partitions. Under this analogy each card of the deck will correspond to a leaf of the tree determining a threshold formula.

Definition 9.29 (Cuts and shuffles)**.** A *cut* of a vector of variables $(a_1, \dots, a_n)$ is a pair $\{(a_1, \dots, a_m), (a_{m+1}, \dots, a_n)\}$. A *riffle shuffle*, or simply *shuffle*, of length n is a string $\sigma \in \{0, 1\}^n$.

For a vector $\boldsymbol{a}$ and shuffle σ of length $n = |\boldsymbol{a}|$ we write $\sigma(\boldsymbol{a})$ to denote the following action of σ on $\boldsymbol{a}$: let Σ_i denote the number of 1s in $\sigma_1 \cdots \sigma_i$, so that $i - \Sigma_i$ is the number of 0s in $\sigma_1 \cdots \sigma_i$ and $k = \Sigma_n$ is the number of 1s in σ; we give a componentwise definition of $\sigma(\boldsymbol{a})$:

$$(\sigma(\boldsymbol{a}))_i = \begin{cases} a_{i-\Sigma_i} & \sigma_i = 0 \\ a_{n-k+\Sigma_i} & \sigma_i = 1 \end{cases}$$

In the above definition, one should think of the 0s and 1s indicating whether a card is dropped from the left or right partition of the deck, with the cut determined by the number of 0s (or equivalently 1s).

Lemma 9.30 (Cutting)**.** *For any tree T and cut $\{\boldsymbol{a}, \boldsymbol{b}\}$ there are trees S_0, S_1 with $d(S_0), d(S_1) \leq d(T)$ such that there are monotone derivations*

$$\begin{array}{c} \mathsf{th}_k^T(\boldsymbol{a}, \boldsymbol{b}) \\ \Big\Vert \\ \bigvee\limits_{i+j=k} \mathsf{th}_i^{S_0}(\boldsymbol{a}) \wedge \mathsf{th}_j^{S_1}(\boldsymbol{b}) \end{array}$$

whose flows have length $O(d(T))$ and width $O(l(T))$.

Proof. By induction on $l(T)$. Without loss of generality assume $\boldsymbol{b}$ is contained entirely in T_1 (otherwise $\boldsymbol{a}$ is contained entirely in T_0 and the argument is symmetric). We

$$
\begin{array}{rc}
 & \mathsf{th}^{S}_{r}(\boldsymbol{w},\boldsymbol{x}) \\
\hline
= & \bigvee_{s+t=r}\left(\begin{array}{c}
\left(\begin{array}{ccc}
\mathsf{th}^{S_0}(\boldsymbol{w}) & & \mathsf{th}^{S_1}(\boldsymbol{x}) \\
\text{cut}\Big\| & \wedge & \text{cut}\Big\| \\
\bigvee_{i+j=s} \mathsf{th}^{S_0'}_{i}(\boldsymbol{a},\boldsymbol{b}^1)\wedge \mathsf{th}^{R_0}_{j}(\boldsymbol{b}^2) & & \bigvee_{k+l=t} \mathsf{th}^{R_1}_{k}(\boldsymbol{c}^1)\wedge \mathsf{th}^{S_1'}_{l}(\boldsymbol{c}^2,\boldsymbol{d})
\end{array}\right) \\
\mathsf{dist}\!\uparrow\Big\| \\
\bigvee_{\substack{i+j=s\\ k+l=t}} \left(\mathsf{th}^{S_0'}_{i}\left(\boldsymbol{a},\boldsymbol{b}^1\right)\wedge \mathsf{th}^{R_0}_{j}\left(\boldsymbol{b}^2\right)\right)\wedge\left(\mathsf{th}^{R_1}_{k}\left(\boldsymbol{c}^1\wedge \mathsf{th}^{S_1'}_{l}\left(\boldsymbol{c}^2,\boldsymbol{d}\right)\right)\right)
\end{array}\right) \\
\hline
= & \bigvee_{s'+t'=r}\left(\begin{array}{c}
\bigvee_{\substack{i+k=s'\\ j+l=t'}} \left(\mathsf{th}^{S_0'}_{i}\left(\boldsymbol{a},\boldsymbol{b}^1\right)\wedge \mathsf{th}^{R_1}_{k}\left(\boldsymbol{c}^1\right)\right)\wedge\left(\mathsf{th}^{R_0}_{j}\left(\boldsymbol{b}^2\right)\wedge \mathsf{th}^{S_1'}_{l}\left(\boldsymbol{c}^2,\boldsymbol{d}\right)\right) \\
\Big\|\mathsf{dist}\!\downarrow \\
\left(\begin{array}{ccc}
\mathsf{th}^{T_0'}_{s'}(\boldsymbol{a},\boldsymbol{b}^1,\boldsymbol{c}^1) & & \mathsf{th}^{T_1'}_{t'}(\boldsymbol{b}^2,\boldsymbol{c}^2,\boldsymbol{d}) \\
IH\Big\| & \wedge & IH\Big\| \\
\mathsf{th}^{T_0}_{s'}(\boldsymbol{y}) & & \mathsf{th}^{T_1}_{t'}(\boldsymbol{z})
\end{array}\right)
\end{array}\right) \\
\hline
= & \mathsf{th}^{T}_{r}(\boldsymbol{y},\boldsymbol{z})
\end{array}
$$

Figure 9-1: Riffle shuffling the inputs of a threshold formula.

construct the following derivation,

$$
\begin{array}{rc}
 & \mathsf{th}^{T}_{r}(\boldsymbol{a},\boldsymbol{b}) \\
\hline
= & \bigvee_{s+t=r}\left(\begin{array}{c}
\left(\mathsf{th}^{T_0}_{s}\left(\boldsymbol{a}^1\right)\wedge \begin{array}{c}
\mathsf{th}^{T_1}_{t}(\boldsymbol{a}^2,\boldsymbol{b}) \\
IH\Big\| \\
\bigvee_{i+j=t} \mathsf{th}^{S_0'}_{i}\left(\boldsymbol{a}^2\right)\wedge \mathsf{th}^{S_1}_{j}\left(\boldsymbol{b}\right)
\end{array}\right) \\
\Big\|\mathsf{dist}\!\uparrow \\
\bigvee_{i+j=t} \mathsf{th}^{T_0}_{s}\left(\boldsymbol{a}^1\right)\wedge \mathsf{th}^{S_0'}_{i}\left(\boldsymbol{a}^2\right)\wedge \mathsf{th}^{S_1}_{j}\left(\boldsymbol{b}\right)
\end{array}\right) \\
\hline
= & \bigvee_{s'+t'=r}\left(\begin{array}{c}
\bigvee_{k+l=s'} \mathsf{th}^{T_0}_{k}\left(\boldsymbol{a}^1\right)\wedge \mathsf{th}^{S_0'}_{l}\left(\boldsymbol{a}^2\right)\wedge \mathsf{th}^{S_1}_{t'}\left(\boldsymbol{b}\right) \\
\Big\|\mathsf{dist}\!\downarrow \\
\left(= \dfrac{\bigvee_{k+l=s'} \mathsf{th}^{T_0}_{k}\left(\boldsymbol{a}^1\right)\wedge \mathsf{th}^{S_0'}_{l}\left(\boldsymbol{a}^2\right)}{\mathsf{th}^{S_0}_{s'}(\boldsymbol{a})}\wedge \mathsf{th}^{S_1}_{t'}(\boldsymbol{b})\right)
\end{array}\right)
\end{array}
$$

where the derivation marked IH is obtained by the inductive hypothesis. $\Box$

Lemma 9.31 (Shuffling)**.** *Let S be a tree and σ a shuffle of length $l(S)$. There is a* *tree T with $d(T) = O(d(S))$ and monotone derivations,*

$$\begin{array}{c} \mathsf{th}_k^S(\boldsymbol{v}) \\ \Vert \\ \mathsf{th}_k^T(\sigma(\boldsymbol{v})) \end{array}$$

whose flows have length $O(d(S)^2)$ and width $O(l(S))$.

Proof. By induction on $l(S)$. We give the inductive step in Fig. 9-1. In the construction we set $\boldsymbol{v} = (\boldsymbol{w}, \boldsymbol{x})$, defined by $l(S_0)$ and $l(S_1)$, and $\sigma(\boldsymbol{v}) = (\boldsymbol{y}, \boldsymbol{z})$. The argument is analogous to the one in Lemma 9.14, with derivations marked 'cut' obtained from Lemma 9.30 and derivations marked IH obtained from the inductive hypothesis.

In particular the cuts are chosen such that $|\boldsymbol{b}^2| = |\boldsymbol{c}^1|$ and so that there are shuffles σ_1, σ_2 with $\sigma(\boldsymbol{v}) = (\sigma_1(\boldsymbol{a}, \boldsymbol{b}^1, \boldsymbol{c}^1), \sigma_2(\boldsymbol{b}^2, \boldsymbol{c}^2, \boldsymbol{d}))$. Such a choice exists (and is unique) by the intermediate value theorem. □

Theorem 9.32 (Merge sort)**.** *For any tree S and permutation π on $\{1, \dots, l(S)\}$ there is a tree T with $d(T) = O(d(S))$ and monotone derivations,*

$$\begin{array}{c} \mathsf{th}_k^S(a_{i\pi})_{i=1}^n \\ \Vert \\ \mathsf{th}_k^T(a_i)_{i=1}^n \end{array}$$

whose flows have length $O(d(S)^3)$ and width $O(l(S))$.

Proof. By induction on $l(S) = l(T)$. We construct the following derivation,

$$\begin{array}{c} \mathsf{th}_r^S(a_{i\pi})_{i=1}^n \\ \hline \displaystyle\bigvee_{s+t=r} \left(\begin{array}{ccc} \mathsf{th}_s^{S_0}(a_{i\pi})_{i=1}^m & & \mathsf{th}_t^{S_1}(a_{i\pi})_{i=m+1}^n \\ {\scriptstyle IH}\Vert & \wedge & {\scriptstyle IH}\Vert \\ \mathsf{th}_s^{T_0'}(\boldsymbol{a}^1) & & \mathsf{th}_t^{T_1'}(\boldsymbol{a}^2) \end{array} \right) \\ {\scriptstyle \text{shuffle}}\Vert \\ \mathsf{th}_r^T(a_i)_{i=1}^n \end{array}$$

where the derivations marked IH are obtained from the inductive hypothesis, sorting the inputs of the left and right subtrees of S to vectors $\boldsymbol{a}^1$ and $\boldsymbol{a}^2$ resp., and the derivation marked 'shuffle', obtained from Lemma 9.31, carries out the unique shuffle on $(\boldsymbol{a}^1, \boldsymbol{a}^2)$ resulting in a completely sorted vector. □

Proposition 9.33 (Repartitionings)**.** *For trees S and T with the same number of leaves, there are monotone derivations,*

$$\begin{array}{c} \mathsf{th}_k^S(\boldsymbol{a}) \\ \| \\ \mathsf{th}_k^T(\boldsymbol{a}) \end{array}$$

whose flows have length $O(d(S)^2)$ and width $O(l(S))$.

Proof. By induction on $l(S) = l(T)$. Let $\{\boldsymbol{b}, \boldsymbol{c}\}$ be the cut of $\boldsymbol{a}$ such that $|\boldsymbol{b}| = l(T_0)$ and $|\boldsymbol{c}| = l(T_1)$. We construct the following derivation,

$$\begin{array}{c} \mathsf{th}_k^S(\boldsymbol{a}) \\ {\scriptstyle \text{cut}}\Big\| \\ = \dfrac{\displaystyle\bigvee_{i+j=k} \left(\begin{array}{c} \mathsf{th}_i^{R_0}(\boldsymbol{b}) \\ {\scriptstyle IH}\Big\| \\ \mathsf{th}_i^{T_0}(\boldsymbol{b}) \end{array} \wedge \begin{array}{c} \mathsf{th}_j^{R_1}(\boldsymbol{c}) \\ {\scriptstyle IH}\Big\| \\ \mathsf{th}_j^{T_1}(\boldsymbol{c}) \end{array} \right)}{\mathsf{th}_k^T(\boldsymbol{b}, \boldsymbol{c})} \end{array}$$

where the derivation marked ‘cut’ is obtained from Lemma 9.30 and the derivations marked IH are obtained from the inductive hypothesis. □

Theorem 9.34. *For any tree T and permutation π on $\{1, \ldots, l(T)\}$, there are normal derivations,*

$$\begin{array}{c} \mathsf{th}_k^T(a_i)_{i=1}^n \\ \Big\| \\ \mathsf{th}_k^T(a_{i\pi})_{i=1}^n \end{array}$$

of size $l(T)^{O(d(T)^3)}$.

Proof. By Thm. 9.32 and Prop. 9.33 we have such derivations whose flows have length $O(d(T)^3)$ and width $O(l(T))$, whence normal derivations of the required size can be constructed by Thm. 6.22 and Props. 6.24 and 9.10. □

Corollary 9.35. *There are quasipolynomial-size normal derivations from $\mathsf{th}_k^n(a_i)$ to $\mathsf{th}_k^n(a_{i\pi})$, for any permutation π on $\{1, \ldots, n\}$.*

9.4 Further results and applications

In this section we apply the methods and results of the previous two sections to some other combinatorial principles in KS.

9.4.1 Generalised pigeonhole principle

If there are 45 hats that are either red or green, then there must be 23 of the same colour. This exemplifies a generalisation of the pigeonhole principle where sufficiently many pigeons may guarantee more than two in some hole [Dij91]. If $k+1$ pigeons in some hole are required then $nk+1$ pigeons are necessary, so this principle can be encoded as follows:

$$\bigwedge_{i=0}^{nk+1} \bigvee_{j=1}^{n} a_{ij} \to \bigvee_{i_r < i_{r+1}} \bigwedge_{r=1}^{k+1} a_{i_r j}$$

This formula has size $O(n^{k+2})$, polynomial for fixed k. If, however k is large relative to n, e.g. $n/2$ or $\sqrt{n}$, then one can always express the right hand side using threshold formulae to obtain an encoding of quasipolynomial-size.

It is not difficult to see that our proofs of PHP_n can easily be generalised to this class of tautologies, by the same arguments as in Sect. 9.2.3.

9.4.2 Parity principle

The *parity principle* states that one cannot partition an odd-size set into pairs, and is encoded by the following propositional tautologies,

$$\mathsf{PAR}_n \quad : \quad \bigwedge_{i=0}^{2n} \bigvee_{j \neq i} a_{\{i,j\}} \to \bigvee_{j \neq i > i' \neq j} a_{\{i,j\}} \wedge a_{\{i',j\}}$$

where $a_{\{i,j\}}$ should be interpreted as "element i is paired with element j".

These tautologies have similar structure to PHP_n, but in many proof systems these tautologies are in fact *harder* to prove. For example, in bounded-depth Hilbert-Frege systems it is known that one can efficiently derive PHP_n from PAR_n but not vice-versa [Ajt90] [BP96].

However, in KS, we can construct quasipolynomial-size proofs of PAR_n using similar methods to those for PHP_n, and we give an outline of these constructions in this subsection. Again, this highlights the difference in behaviour between systems arising from restricting formula-depth and from restricting negation and structural interactions.

We omit proofs corresponding to basic properties of threshold functions, since they are fairly routine inductions of which the previous two sections have given many examples, and also often do not specify precise orderings of variables or tree-structures of a threshold formulae, since these can all be reduced to any other in quasipolynomial time, by the results of Sect. 9.3.

Let LPAR_n and RPAR_n denote the left and right hand sides of PAR_n respectively.

By a similar argument to Prop. 9.19 we obtain normal derivations of the following form,

$$\begin{array}{c}\mathsf{LPAR}_n\\ \Vert\\ \mathsf{th}^{2n(2n+1)}_{2n+1}(\boldsymbol{a}^2)\end{array}$$

where $\boldsymbol{a}^2$ is an appropriate sequence of the variables $a_{\{i,j\}}$ in which each variable occurs exactly twice, as in LPAR_n.

Let $(\boldsymbol{a},\boldsymbol{a})$ be a permutation of $\boldsymbol{a}^2$ so that each variable occurs exactly once in $\boldsymbol{a}$. Now we can construct the following derivation,

$$\begin{array}{c}\mathsf{th}^{2n(2n+1)}_{2n+1}(\boldsymbol{a}^2)\\ \text{permute}\Vert\\ \mathsf{th}^{2n(2n+1)}_{2n+1}(\boldsymbol{a},\boldsymbol{a})\\ \text{evaluate}\Vert\\ \mathsf{c}\!\downarrow\dfrac{\mathsf{th}^{n(2n+1)}_{n+1}(\boldsymbol{a})\vee\mathsf{th}^{n(2n+1)}_{n+1}(\boldsymbol{a})}{\mathsf{th}^{n(2n+1)}_{n+1}(\boldsymbol{a})}\end{array}$$

where the derivation marked 'permute' applies the results of Sect. 9.3, namely Cor. 9.35, to permute the arguments of a threshold formula, and the derivation marked 'evaluate' is obtained by Lemma 9.21, setting $r = n$ and $s = n + 1$.

Now notice that, if $n+1$ of the variables $a_{\{i,j\}}$ are true, i.e. we have $n+1$ pairs out of $2n + 1$ variables, we must have some j which is paired with two distinct variables, and this can be realised as derivations,

$$\begin{array}{c}\mathsf{th}^{n(2n+1)}_{n+1}(\boldsymbol{a})\\ \Vert\\ \mathsf{RPAR}_n\end{array}$$

in a similar way to Prop. 9.19.

Chaining all these normal derivations together gives us quasipolynomial-size monotone derivations $\begin{array}{c}\mathsf{LPAR}_n\\ \Vert\\ \mathsf{RPAR}_n\end{array}$ with flows of bounded length, and from here we can construct quasipolynomial-size KS-proofs of PAR_n in the usual way.

9.4.3 Intermediate value theorem

Recall the various threshold formulae constructions from Sect. 5.1, in particular the formulae arising from the identity,

$$\mathsf{TH}^{2n}_{2k}(a_1,\dots,a_{2n}) = \bigvee_{i=1}^{n} \mathsf{TH}^n_k(a_i,\dots,a_{i+n-1}) \wedge \mathsf{TH}^n_k(a_1,\dots,a_{i-1},a_{i+n},\dots,a_{2n})$$

which we will denote Θ^n_k.

We could consider the class of tautologies,

$$\mathsf{IVT}^n_k \quad : \quad \mathsf{th}^n_k(\boldsymbol{a}) \to \Theta^n_k(\boldsymbol{a})$$

as an encoding of the intermediate value theorem, in the sense that it expresses the statement "in any circle of booleans there is some semicircle and its complement that each contain half of the 1s", which we would naturally verify using the intermediate value theorem.

Initial attempts to generalise the methods of this and previous sections to obtain small normal proofs of this implication have failed, and I do not see any naïve way in which they can be adapted. Problems arise from the fact that the identity above does not uniformly split its arguments in the same way as for th^n_k, but rather relies on strong logical dependencies between its disjunctions, dictated by the intermediate value theorem.

Trying to 'unwind' these dependencies in order to carry out some sort of induction argument unfortunately requires us to 'rewind' them, and so each inductive step seemingly requires us to apply derivations obtained by the inductive hypothesis twice in *series*, doubling the length at each stage and resulting in derivations whose flows have polynomial length. Incidentally, such an approach does succeed in producing quasipolynomial-size monotone derivations, and so also KS^+-proofs, for IVT^n_k.

If there were some superpolynomial separation between KS and KS^+, we would suggest that analysing proofs of IVT^n_k might be a good starting point. If nothing else, it would be interesting to understand the limits of the techniques we have presented in this chapter.

Part V

Conclusions

Chapter 10

Final remarks

In this dissertation we studied the proof complexity of propositional deep inference systems for classical logic. Our main focus has been the strength of KS, for which we developed tools to reason about proof complexity and applied to obtain original results. Nonetheless, the outstanding unresolved problem arising from this work is the position of KS in the proof complexity heirarchy.

Open problem 10.1. *Show that* KS *(quasi)polynomially simulates Hilbert-Frege systems, or that it is nontrivially separated from* KS^+*.*

Towards a simulation, the tools from Chapt. 6 have already proved fruitful in yielding positive complexity results for KS and, perhaps together with some other insights, could provide the right setting in which to construct some appropriate normalisation procedure.

Towards a separation, we have singled out a class of tautologies, IVT^n_k, in Sect. 9.4 for which the methods we have developed do not readily apply, yet which have quasipolynomial-size proofs in KS^+. If these did witness some superpolynomial separation between the two systems, results from Chapt. 7 might be helpful in proving this. Determining whether a KS-proof of a certain size exists can be reduced to deciding whether an MS path exists between two terms. Since these terms are governed by a single parameter k, the design of appropriate measures, e.g. in Thm. 7.1, could be useful in showing that no such paths exist unless k is large.

While this work already provides some small contribution to proof complexity, by bridging the gap in a novel way to Hilbert-Frege systems, we have also made other independent observations in this direction, namely the reductions from Hilbert-Frege provability in Cor. 4.10 and Cor. 7.20.

Perhaps most prominently we point out that, since this dissertation was written, the proof structure of the pigeonhole principle presented in 9.2 was adapted to yield

$n^{O(\log\log n)}$-size monotone proofs of quite strong versions of the weak pigeonhole principle, namely with n holes and $(1+\varepsilon)n$ pigeons for $\varepsilon = 1/\operatorname{polylog} n$ [Das14]. This improves on the bound of $n^{O(\log n)}$ inherited from proofs of the unrestricted pigeonhole principle in [AGG00] and is the first time that considerations in the complexity of deep inference have lead to improvements in more mainstream systems in proof complexity. We tentatively propose that this suggests that ongoing research into the complexity of deep inference proofs might yield further contributions to the wider area.

There are many relevant problems that we have not considered, one being the complexity of systems with added compression mechanisms, e.g. extension and substitution; we would refer the reader to [BG09b] and [Str09] for related results on deep inference systems augmented with such features. We have also not addressed the complexity of predicate logic or logics besides classical logic in deep inference. It would be interesting to study normalisation in these settings, in particular to examine the effects of various structural and logical features on the complexity of such procedures.

There are many further directions in which the ideas of this dissertation could be extended, as we have mentioned throughout. However, we believe that it would be particularly worthwhile to develop the techniques presented in Part III, on the structural and logical fragments of propositional systems, and we discuss some possible developments to this end.

The rewriting systems on atomic flows that we presented were independent of the logical fragment of deep inference; arguably, this might restrict the range of flow rewriting rules available to us. Can we develop rewriting systems whose soundness is fundamentally reliant on the logical behaviour of switch and medial, rather than just their linearity? Furthermore, could we obtain more powerful rewriting systems by adding certain logical information to flows, for example by specifying medial steps between structural inferences? We note that some of these ideas are currently being pursued by various researchers in the deep inference community.

Can the study of linear inferences shed some nontrivial insights into questions of proof complexity, and in turn coNP vs. NP? It seems we can develop an analogous 'proof theory' for linear inferences, although the difficulties of analogous questions do not seem to match up. For example, while it is simple to construct a complete system for TAUT, as a basis of sound inferences under rewriting, the same is not known to exist for the set of linear inferences. This is arguably because the complexity of linear systems is determined by the length of derivations, unlike traditional proof systems whose complexities can exhibit nontrivial interplays between length and width. Nonetheless, certain sets of linear inferences are fundamentally related to certain proof systems in ways that are complexity-sensitive, e.g. MS to KS. It would be interesting to examine whether such a set, not necessarily finite, could be associated to arbitrary proof systems, for example by encoding the soundness of a proof system as a class of linear inferences. This is similar to adding the linear encodings of the pigeonhole principle to KS, as we mentioned in Sect. 7.4, to obtain a system that has polynomial-size proofs of PHP_n.

Bibliography

[AGG00] Albert Atserias, Nicola Galesi, and Ricard Gavalda. Monotone proofs of the pigeon hole principle. 2000.

[AGP02] Albert Atserias, Nicola Galesi, and Pavel Pudlák. Monotone simulations of non-monotone proofs. *Journal of Computer and System Sciences*, 65(4):626–638, 2002.

[Ajt90] M. Ajtai. Parity and the pigeonhole principle. In SamuelR. Buss and PhilipJ. Scott, editors, *Feasible Mathematics*, volume 9 of *Progress in Computer Science and Applied Logic*, pages 1–24. Birkhuser Boston, 1990.

[AKS83] M. Ajtai, J. Komlós, and E. Szemerédi. An 0(n log n) sorting network. In *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, STOC '83, pages 1–9, New York, NY, USA, 1983. ACM.

[BG09a] Paola Bruscoli and Alessio Guglielmi. On analyticity in deep inference. `http://cs.bath.ac.uk/ag/p/ADI.pdf`, 2009.

[BG09b] Paola Bruscoli and Alessio Guglielmi. On the proof complexity of deep inference. *ACM Transactions on Computational Logic*, 10(2):1–34, 2009. Article 14. `http://cs.bath.ac.uk/ag/p/PrComplDI.pdf`.

[BGGP10] Paola Bruscoli, Alessio Guglielmi, Tom Gundersen, and Michel Parigot. A quasipolynomial cut-elimination procedure in deep inference via atomic flows and threshold formulae. In Edmund M. Clarke and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning (LPAR-16)*, volume 6355 of *Lecture Notes in Computer Science*, pages 136–153. Springer-Verlag, 2010. `http://cs.bath.ac.uk/ag/p/QPNDI.pdf`.

[BL05] Kai Brünnler and Stéphane Lengrand. On two forms of bureaucracy in derivations. In Paola Bruscoli, François Lamarche, and Charles Stewart, editors, *Structures and Deduction*, pages 69–80. Technische Universität Dresden, 2005. ICALP Workshop. ISSN 1430-211X. `http://www.iam.unibe.ch/~kai/Papers/sd05.pdf`.

[BP96] Paul Beame and Toniann Pitassi. An exponential separation between the parity principle and the pigeonhole principle. pages 195–228, 1996.

[BPR97] ML Bonet, T. Pitassi, and R. Raz. No feasible interpolation for tc0-frege proofs. In *focs*, page 254. Published by the IEEE Computer Society, 1997.

[Brü03] Kai Brünnler. Two restrictions on contraction. *Logic Journal of the IGPL*, 11(5):525–529, 2003. `http://www.iam.unibe.ch/~kai/Papers/RestContr.pdf`.

[Brü04] Kai Brünnler. *Deep Inference and Symmetry in Classical Proofs.* Logos Verlag, Berlin, 2004. `http://www.iam.unibe.ch/~kai/Papers/phd.pdf`.

[Brü06] Kai Brünnler. Deep inference and its normal form of derivations. In Arnold Beckmann, Ulrich Berger, Benedikt Löwe, and John V. Tucker, editors, *Computability in Europe 2006*, volume 3988 of *Lecture Notes in Computer Science*, pages 65–74. Springer-Verlag, July 2006. `http://www.iam.unibe.ch/~kai/Papers/n.pdf`.

[BT01] Kai Brünnler and Alwen Fernanto Tiu. A local system for classical logic. Technical Report WV-01-02, Technische Universität Dresden, 2001. `http://iccl.tu-dresden.de/~kai/LocalClassicalLogic-tr.pdf`.

[Bus87] Samuel R. Buss. Polynomial size proofs of the propositional pigeonhole principle. *Journal of Symbolic Logic*, 52(4):916–927, 1987.

[Bus11] Samuel R. Buss. Towards NP P via proof complexity and search. *Annals of Pure and Applied Logic*, 2011.

[Coo71] Stephen A Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158. ACM, 1971.

[Coo75] Stephen A. Cook. Feasibly constructive proofs and the propositional calculus (preliminary version). In *Proceedings of seventh annual ACM symposium on Theory of computing*, STOC '75, pages 83–97, New York, NY, USA, 1975. ACM.

[CR74a] Stephen Cook and Robert Reckhow. On the lengths of proofs in the propositional calculus. In *Proceedings of the 6th annual ACM Symposium on Theory of Computing*, pages 135–148. ACM Press, 1974.

[CR74b] Stephen Cook and Robert Reckhow. On the lengths of proofs in the propositional calculus (preliminary version). In *Proceedings of the 6th annual ACM Symposium on Theory of Computing*, pages 135–148. ACM Press, 1974.

[D'A92] Marcello D'Agostino. Are tableaux an improvement on truth-tables? *Journal of Logic, Language and Information*, 1:235–252, 1992. 10.1007/BF00156916.

[Das11] Anupam Das. On the proof complexity of cut-free bounded deep inference. In Kai Brünnler and George Metcalfe, editors, *Tableaux 2011*, volume 6793 of *Lecture Notes in Artificial Intelligence*, pages 134–148. Springer-Verlag, 2011. `http://www.anupamdas.com/items/PrCompII/ProofComplexityBoundedDI.pdf`.

[Das12] Anupam Das. Complexity of deep inference via atomic flows. In S. Barry Cooper, Anuj Dawar, and Benedikt Löwe, editors, *Computability in Europe*, volume 7318 of *Lecture Notes in Computer Science*, pages 139–150. Springer-Verlag, 2012. `http://www.anupamdas.com/items/RelComp/RelComp.pdf`.

[Das13] Anupam Das. Rewriting with linear inferences in propositional logic. In Femke van Raamsdonk, editor, *RTA*, volume 21 of *LIPIcs*, pages 158–173. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2013.

[Das14] Anupam Das. On the pigeonhole and related principles in deep inference and monotone systems. In *Joint Meeting of the 23rd EACSL Annual Conference on Computer Science Logic (CSL) and the 29th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2014.

[Dij91] Edsger W. Dijkstra. The undeserved status of the pigeon-hole principle. March 1991.

[Gen35] Gerhard Gentzen. Untersuchungen ber das logische schlieen. i. *Mathematische Zeitschrift*, 39(1):176–210, 1935.

[GG08] Alessio Guglielmi and Tom Gundersen. Normalisation control in deep inference via atomic flows. *Logical Methods in Computer Science*, 4(1:9):1–36, 2008. `http://www.lmcs-online.org/ojs/viewarticle.php?id=341`.

[GGP10] Alessio Guglielmi, Tom Gundersen, and Michel Parigot. A proof calculus which reduces syntactic bureaucracy. In Christopher Lynch, editor, *RTA 2010*, volume 6 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 135–150. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2010. `http://drops.dagstuhl.de/opus/volltexte/2010/2649`.

[GGS10] Alessio Guglielmi, Tom Gundersen, and Lutz Straßburger. Breaking paths in atomic flows for classical logic. In Jean-Pierre Jouannaud, editor, *25th Annual IEEE Symposium on Logic in Computer Science*, pages 284–293. IEEE, 2010. `http://www.lix.polytechnique.fr/~lutz/papers/AFII.pdf`.

[Gir87] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[Gun09] Tom Gundersen. *A General View of Normalisation Through Atomic Flows.* PhD thesis, University of Bath, 2009.

[Gur77] VA Gurvich. Repetition-free boolean functions. *Uspekhi Matematicheskikh Nauk*, 32(1):183–184, 1977.

[Jeř09] Emil Jeřábek. Proof complexity of the cut-free calculus of structures. *Journal of Logic and Computation*, 19(2):323–339, 2009. `http://www.math.cas.cz/~jerabek/papers/cos.pdf`.

[Jeř11] Emil Jeřábek. A sorting network in bounded arithmetic. *Annals of Pure and Applied Logic*, 162(4):341–355, 2011.

[Jeř12] Emil Jeřábek. Proofs with monotone cuts. *Mathematical Logic Quarterly*, 58(3):177–187, 2012.

[Klo92] Jan Willem Klop. Term rewriting systems. In S. Abramsky, D. Gabbay, and T. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 2, pages 1–116. Oxford University Press, 1992.

[KPW95] 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 & Algorithms*, 7(1):15–39, 1995.

[Kra95] Jan Krajíček. *Bounded arithmetic, propositional logic, and complexity theory.* Cambridge University Press, New York, NY, USA, 1995.

[PBI93] Toniann Pitassi, Paul Beame, and Russell Impagliazzo. Exponential lower bounds for the pigeonhole principle. *Computational Complexity*, 3:97–140, 1993. 10.1007/BF01200117.

[PW81] J.B. Paris and A.J. Wilkie. δ_0 sets and induction. *Open Days in Model Theory and Set Theory, W. Guzicki, W. Marek, A. Pelc, and C. Rauszer, eds*, pages 237–248, 1981.

[Raz02] AlexanderA. Razborov. Proof complexity of pigeonhole principles. In Werner Kuich, Grzegorz Rozenberg, and Arto Salomaa, editors, *Developments in Language Theory*, volume 2295 of *Lecture Notes in Computer Science*, pages 100–116. Springer Berlin Heidelberg, 2002.

[Str07] Lutz Straßburger. A characterisation of medial as rewriting rule. In Franz Baader, editor, *RTA 2007*, volume 4533 of *Lecture Notes in Computer Science*, pages 344–358. Springer-Verlag, 2007. `http://www.lix.polytechnique.fr/~lutz/papers/CharMedial.pdf`.

[Str09] Lutz Straßburger. Extension without cut. Submitted. `http://www.lix.polytechnique.fr/~lutz/papers/psppp.pdf`, 2009.

[Thi03] Rüdiger Thiele. Hilbert's twenty-fourth problem. *American Mathematical Monthly*, 110:1–24, 2003.

[TS96] A.S. Troelstra and H. Schwichtenberg. *Basic Proof Theory*, volume 43 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1996.

[Val84] L.G Valiant. Short monotone formulae for the majority function. *Journal of Algorithms*, 5(3):363 – 366, 1984.

[Š12] Alvin Šipraga. An automated search of linear inference rules. `http://arcturus.su/mimir/autolininf.pdf`, 2012.