

QUASIPOLYNOMIAL NORMALISATION IN DEEP INFERENCE VIA ATOMIC FLOWS AND THRESHOLD FORMULAE

PAOLA BRUSCOLI, ALESSIO GUGLIELMI, TOM GUNDERSEN, AND MICHEL PARIGOT

ABSTRACT. Jeřábek showed that cuts in classical propositional logic proofs in deep inference can be eliminated in quasipolynomial time. The proof is indirect and it relies on a result of Atserias, Galesi and Pudlák about monotone sequent calculus and a correspondence between that system and cut-free deep-inference proofs. In this paper we give a direct proof of Jeřábek’s result: we give a quasipolynomial-time cut-elimination procedure for classical propositional logic in deep inference. The main new ingredient is the use of a computational trace of deep-inference proofs called atomic flows, which are both very simple (they only trace structural rules and forget logical rules) and strong enough to faithfully represent the cut-elimination procedure.

1. INTRODUCTION

Deep inference is a proof-theoretic methodology where proofs can be freely composed by the logical operators, instead of having a rigid formula-directed tree structure, as in Gentzen proof theory [Gug07, BT01, Brü04, GGP10]. As a result, inference rules apply arbitrarily deep inside formulae, contrary to traditional proof systems such as natural deduction and the sequent calculus, where inference rules only deal with the outermost structure of formulae. This induces a new symmetry, which can be exploited for achieving locality of inference rules, and which is not available with Gentzen methods. Locality, in turn, makes it possible to use new methods, often with a geometric flavour, in the normalisation theory of proof systems.

The greater freedom in composing proofs of deep inference is both a source of immediate technical difficulty and of new powerful proof-theoretic methods. A general methodology allows us to design deep-inference proof systems having more symmetries and finer structural properties than the sequent calculus does. For instance, cut and identity become really dual of each other, whereas they only are morally so in the sequent calculus, and all the structural rules can be reduced to their atomic form, whereas this does not hold in the sequent calculus, for example in the case of the contraction inference rule. In deep inference, the cut rule is more general than its counterpart in the sequent calculus, and makes it possible to obtain a broader range of dynamics in normalisation procedures. However, despite the sequent calculus systems and their normalisation procedures being special cases of deep inference systems and procedures, cut elimination in deep inference still guarantees consistency and the trivial turning of proof systems into algorithms for proof search.

Date: February 9, 2016.

Bruscoli and Guglielmi have been supported by EPSRC grants EP/E042805/1 *Complexity and Non-determinism in Deep Inference* and EP/K018868/1 *Efficient and Natural Proof Systems*, and by an ANR Senior Chaire d’Excellence titled *Identity and Geometric Essence of Proofs*.

Gundersen has been supported by an *Overseas Research Studentship* and a *Research Studentship*, both of the University of Bath, and by an ANR Senior Chaire d’Excellence titled *Identity and Geometric Essence of Proofs*.

Parigot has been supported by *Project INFER—Theory and Application of Deep Inference* of the *Agence Nationale de la Recherche*.

Gundersen and Parigot are supported by *Project STRUCTURAL—Structural and Computational Proof Theory* of the *Agence Nationale de la Recherche*.

Sixteen years of research have guaranteed that all usual logics have deep-inference proof systems enjoying cut elimination (see [Gug] for a complete overview). The traditional methods of cut elimination of the sequent calculus can be adapted to a large extent to deep inference, despite having to cope with a higher generality, but new methods are also achievable. The standard proof system for propositional classical logic in deep inference is system SKS, and its cut elimination has been achieved in several different ways [BT01, Brü04, GG08], all requiring at worst exponential time in the size of the proof to be normalised.

A few years ago Jeřábek showed that cut elimination in SKS proofs can be done in quasipolynomial time [Jeř09], more specifically in time $n^{O(\log n)}$. The result is surprising because all known cut-elimination methods for classical-logic proof systems require exponential time, in particular for Gentzen’s sequent calculus. Jeřábek obtained his result by relying on a construction over threshold functions by Atserias, Galesi and Pudlák, in the monotone sequent calculus [AGP02].

Jeřábek’s technique is indirect because normalisation is performed over proofs in the sequent calculus, which are, in turn, related to deep-inference ones by polynomial simulations, originally studied in [Brü06] and [BG09].

In this paper we give a direct proof of Jeřábek’s result: that is, we give a quasipolynomial-time cut-elimination procedure in propositional-logic deep inference, which, in addition to being internal, has a strong computational flavour. Our proof uses two ingredients:

- (1) an adaptation of Atserias-Galesi-Pudlák technique to deep inference, which simplifies the technicalities associated with the use of threshold functions; in particular, the formulae and derivations that we adopt are smaller and structurally simpler than those in [AGP02];
- (2) a recently introduced graphic formalism, tracing atoms in deep-inference proofs, called ‘atomic flows’ [GG08].

Atomic flows, which can be considered specialised Buss flow graphs [Bus91], play a major role in designing and controlling the cut elimination procedure presented in this paper. Atomic flows are very simple (they only trace structural rules and forget logical rules) but they are strong enough to faithfully represent cut elimination [GG08, GGS10] and to relate several formalisms regarding their proof complexity [Das12, Das14]. Atomic flows contribute to the overall clarification of this highly technical matter, by reducing our dependency on syntax. The techniques developed via atomic flows tolerate variations in the proof system specification. In fact, their geometric nature makes them largely independent of syntax, provided that certain linearity conditions are respected (and this is usually achievable in deep inference).

The paper is self-contained. Sections 2 and 3 are devoted, respectively, to the necessary background on deep inference and atomic flows. Threshold functions and formulae are introduced in Section 5.

We normalise proofs in two steps, each of which has a dedicated section in the paper:

- (1) We transform any given proof into what we call its ‘simple form’. No use is made of threshold formulae and no significant proof complexity is introduced. This is presented in Section 4, mostly an exercise on deep inference and atomic flows.
- (2) In Section 6, we show the cut elimination step, starting from proofs in simple form. Here, threshold formulae play a major role.

Normalisation can be taken one step further, by removing the instances of the only inference rule left that is not analytic in the deep-inference sense, *viz.* coweakening. This is performed by a simple and standard deep-inference procedure in Section 7.

Section 8 concludes the paper with comments on future research directions.

Parts of this paper were presented at LPAR 16 [BGGP10] and some appear in [Gun09]. Recently, threshold functions have been used in [Das14] to build quasipolynomial size

cut-free deep-inference proofs of the propositional pigeonhole principle that, crucially, do not use cocontraction, which is a form of dagness.

2. PROPOSITIONAL LOGIC IN DEEP INFERENCE

Inside the deep-inference methodology we can define several formalisms, *i.e.* general prescriptions on how to design proof systems, in the same sense as the sequent calculus and natural deduction are formalisms in Gentzen-style proof theory (where the structure of proofs is determined by the tree structure of the formulae they prove).

The first, and conceptually simpler, formalism that has been defined in deep inference is called the *calculus of structures*, or *CoS* [Gug07]. Another deep-inference formalism has later been introduced in [GGP10], called *open deduction*. Open deduction is more general than CoS, in the sense that every CoS derivation is also an open-deduction derivation. On the other hand, every open-deduction derivation can be transformed into a CoS derivation by a straightforward transformation that essentially amounts to interleaving derivations. The cost of this transformation is at most quadratic in the size of the original open-deduction derivation; therefore, from the point of view of complexity, CoS and open deduction are equivalent.

CoS and open deduction are equivalent also from the point of view of proof theory, because the two formalisms are just two different notations for derivations of the same nature, and so every derivation transformation that can be performed in one formalism can also be performed in the other. In this paper we will adopt the open-deduction notation, especially because it is more efficient for the reader. However, given that most of the literature in deep inference adopts the CoS notation, which is more similar to the traditional Gentzen syntax, we will present both styles in this section.

The standard proof system of propositional logic in deep inference is called SKS. The basic proof-complexity properties of SKS, and so of propositional logic in deep inference, have been studied in [BG09] (which also could be used as an introduction to SKS). Those properties are:

- SKS is polynomially equivalent to Frege proof systems.
- SKS can be extended with Tseitin's extension and substitution, and the proof systems so obtained are polynomially equivalent to Frege proof systems augmented with extension and substitution.
- Cut-free SKS polynomially simulates cut-free Gentzen proof systems for propositional logic, but the converse does not hold: in fact, Statman's tautologies admit polynomial proofs in cut-free SKS but only exponential ones in cut-free Gentzen [Sta78].

We now quickly introduce all the necessary notions. An excellent and more relaxed introduction to SKS in CoS and its basic properties is [Brü04].

Formulae, denoted by A, B, C and D are freely built from: *units*, f (false), t (true); *atoms*, denoted by a, b, c, d and e ; *disjunction* and *conjunction*, $[A \vee B]$ and $(A \wedge B)$. The different brackets have the only purpose of improving legibility; we usually omit external brackets of formulae, and sometimes we omit superfluous brackets under associativity. On the set of atoms a (non-identical) involution $\bar{\cdot}$ is defined and called *negation*; a and \bar{a} are *dual* atoms. We denote *contexts*, *i.e.*, formulae with a hole, by $K\{ \}$ and $H\{ \}$; for example, if $K\{a\}$ is $b \wedge [a \vee c]$, then $K\{ \}$ is $b \wedge [\{ \} \vee c]$, $K\{b\}$ is $b \wedge [b \vee c]$ and $K\{a \wedge d\}$ is $b \wedge [(a \wedge d) \vee c]$.

Note that negation is only defined for atoms, and this is not a limitation because, thanks to De Morgan laws, negation can always be 'pushed to' atoms. Also, note that there are no negative or positive atoms in an absolute sense; we can only say that if we arbitrarily consider \bar{a} positive, then a must be negative, for example.

For both CoS and open deduction an (*inference*) *rule* ρ is an expression $\rho \frac{A}{B}$, where the formulae A and B are called *premiss* and *conclusion*, respectively; an *instance* of that rule is an expression $\rho \frac{C}{D}$, where C and D are instances of A and B .

System SKS is a proof system, common to CoS and open deduction, defined by the following *structural* inference rules:

$$\begin{array}{ccc}
 \text{ai}\downarrow \frac{t}{a \vee \bar{a}} & \text{aw}\downarrow \frac{f}{a} & \text{ac}\downarrow \frac{a \vee a}{a} \\
 \textit{identity} & \textit{weakening} & \textit{contraction} \\
 \\
 \text{ai}\uparrow \frac{a \wedge \bar{a}}{f} & \text{aw}\uparrow \frac{a}{t} & \text{ac}\uparrow \frac{a}{a \wedge a} \\
 \textit{cut} & \textit{coweakening} & \textit{cocontraction}
 \end{array} ,$$

and by the following two *logical* inference rules:

$$\begin{array}{cc}
 \frac{A \wedge [B \vee C]}{(A \wedge B) \vee C} & \frac{(A \wedge B) \vee (C \wedge D)}{[A \vee C] \wedge [B \vee D]} \\
 \textit{switch} & \textit{medial}
 \end{array} .$$

A *cut-free* derivation is a derivation where $\text{ai}\uparrow$ is not used, *i.e.*, a derivation in $\text{SKS} \setminus \{\text{ai}\uparrow\}$.

In addition to these rules, there is a rule $= \frac{C}{D}$, such that C and D are opposite sides in one of the following equations:

$$(1) \quad \begin{array}{cc}
 A \vee B = B \vee A & A \vee f = A \\
 A \wedge B = B \wedge A & A \wedge t = A \\
 [A \vee B] \vee C = A \vee [B \vee C] & t \vee t = t \\
 (A \wedge B) \wedge C = A \wedge (B \wedge C) & f \wedge f = f
 \end{array} .$$

We do not always show the instances of rule $=$, and when we do show them, we gather several contiguous instances into one. We consider the $=$ rule as implicitly present in all systems. The equality relation $=$ on formulae is defined by closing the equations in (1) by reflexivity, symmetry, transitivity and by stipulating that $A = B$ implies $K\{A\} = K\{B\}$; to indicate literal equality of the formulae A and B we adopt the notation $A \equiv B$.

We now define both styles of derivations, CoS and open deduction. The difference is in the way we compose instances of rules: in CoS we only allow inferences to compose vertically, in chains similar to sequent calculus proofs made only of one-premiss rule instances. In open deduction instead, derivations can be composed by the same connectives that formulae are made of. For simplicity, we give here a definition of open-deduction derivation that is limited to our purposes in this paper, and is not the most general.

In CoS, a rule instance $\rho \frac{C}{D}$ generates an (*inference*) *step* $\rho \frac{K\{C\}}{K\{D\}}$, for each context $K\{ \}$.

A CoS *derivation* from (*premiss*) A to (*conclusion*) B is a chain of inference steps with A at the top and B at the bottom. A derivation can be denoted by

$$\begin{array}{c}
 A \\
 \Phi \parallel_{\mathcal{S}} \\
 B
 \end{array} ,$$

where \mathcal{S} is the name of the proof system or a set of inference rules (we might omit Φ and \mathcal{S}); a *proof*, often denoted by Π , is a derivation with premiss t ; besides Φ , we denote derivations with Ψ . Sometimes we group $n \geq 0$ inference steps of the same rule ρ together into one step, and we label the step with $n \cdot \rho$.

In open deduction, and just for the specific case of propositional logic with \vee , \wedge and negation on atoms, we define the notions of *derivation*, *premiss* and *conclusion* inductively as follows:

- if Φ is a unit or an atom then Φ is a derivation with premiss Φ and conclusion Φ ;
- if Φ is a derivation with premiss A and conclusion B and if Ψ is a derivation with premiss C and conclusion D , then
 - $[\Phi \vee \Psi]$ is a derivation with premiss $[A \vee C]$ and conclusion $[B \vee D]$,
 - $(\Phi \wedge \Psi)$ is a derivation with premiss $(A \wedge C)$ and conclusion $(B \wedge D)$,
 - if $\rho \frac{B}{C}$ is an instance of an inference rule, then $\rho \frac{\Phi}{\Psi}$ is a derivation with premiss A and conclusion D .

We adopt the same conventions as for CoS to denote derivations in open deduction. We omit structural rule names in open-deduction notation.

The first two rows in Figure 2 illustrate with examples all the concepts introduced above. The first row shows three example CoS derivations, and below each of them there is an equivalent derivation in open deduction. An open deduction derivation can be obtained from a CoS one by sharing the contexts in inference steps. Vice versa, a CoS derivation can be obtained from an open deduction one by choosing an order for the chain of inference steps.

Besides SKS, another standard deep-inference system is SKSg, which is the same as SKS, except that it does not contain medial and its structural rules are not restricted to atoms. In particular, we use in this paper the rules

$$w\downarrow \frac{f}{A} \quad , \quad w\uparrow \frac{A}{t} \quad , \quad c\downarrow \frac{A \vee A}{A} \quad \text{and} \quad c\uparrow \frac{A}{A \wedge A} \quad .$$

Clearly, a derivation in SKS is also a derivation in SKSg. It can easily be proved that SKS and all its fragments containing the logical and $=$ rules polynomially simulate, respectively, SKSg and its corresponding fragments [BG09]. For example, $\{s, m, =, ac\downarrow\}$ polynomially simulates $\{s, =, c\downarrow\}$. This allows us to transfer properties from SKS to SKSg; in particular, the main result in this paper, *i.e.*, that SKS proofs can be transformed into cut-free ones in quasipolynomial time, holds also for SKSg proofs. One reason to work with SKS instead of SKSg, as we do in this paper, is that atomicity of rules allows us to use atomic flows more conveniently.

A notable cut-free system is $KS = \{s, m, =, ai\downarrow, aw\downarrow, ac\downarrow\}$, which is complete for propositional logic [BT01, Brü04]; this, of course, entails completeness for all the systems that contain KS, such as SKS.

We can replace instances of nonatomic structural rules by derivations with the same premiss and conclusion, and that only contain atomic structural rules. The price to pay is a quadratic growth in size. This is stated by the following, routine proposition (keep in mind that, from now on, we consider the $=$ rule as implicitly present in all systems). An example is the rightmost upper derivation in Figure 2, which stands for a nonatomic cocontraction.

Proposition 1. *Rule instances of $w\downarrow$, $w\uparrow$, $c\downarrow$ and $c\uparrow$ can be derived in quadratic time by derivations in $\{aw\downarrow\}$, $\{aw\uparrow\}$, $\{m, ac\downarrow\}$ and $\{m, ac\uparrow\}$, respectively.*

Sometimes, we use a nonatomic rule instance to stand for some derivation in SKS that derives that instance, as per Proposition 1.

By $A\{a_1/B_1, \dots, a_b/B_b\}$, we denote the operation of simultaneously substituting formulae B_1, \dots, B_b into all the occurrences of the atoms a_1, \dots, a_b in the formula A , respectively; note that the occurrences of $\bar{a}_1, \dots, \bar{a}_b$ are not automatically substituted. Often, we only substitute certain occurrences of atoms, and these are indicated with superscripts that establish a relation with atomic flows. As a matter of fact, we extend the notion of

substitution to derivations in the natural way, but this requires a certain care. The issue is clarified in Section 3 (see, in particular, Notations 2 and 4 and Proposition 3).

The *size* $|A|$ of a formula A , and the *size* $|\Phi|$ of a derivation Φ , is the number of unit and atom occurrences appearing in it. The size of CoS derivations is, obviously, at most quadratic in the size of the corresponding open-deduction derivations. We use this fact implicitly throughout the paper, and we always measure the CoS size of derivations, even if we show them in open-deduction notation.

3. ATOMIC FLOWS

Atomic flows, which have been introduced in [GG08], are, essentially, specialised Buss flow graphs [Bus91]. They are particular directed graphs associated with SKS derivations: every derivation yields one atomic flow obtained by tracing the atom occurrences in the derivation. Infinitely many derivations correspond to each atomic flow; this suggests that much of the information in a derivation is lost in its associated atomic flow; in particular, there is no information about instances of logical rules, only structural rules play a role. As shown in [GG08, GGS10], it turns out that atomic flows contain sufficient structure to control normalisation procedures, providing in particular induction measures that can be used to ensure termination. Such normalisation procedures require exponential time on the size of the derivation to be normalised. In the present work, we lower the complexity of proof normalisation to quasipolynomial time, but an essential role is played by the complex logical relations of threshold formulae, which are external and independent from the given proof. This means that atomic flows are not sufficient to define the normalisation procedure; however, they still are a very convenient tool for defining and understanding several of its aspects.

We can single out three features of atomic flows that, in general, and not just in this work, help in designing normalisation procedures:

- (1) Atomic flows conveniently express the topological structure of atom occurrences in a proof. This is especially useful for defining a certain ‘simple form’ of proofs (Definition 7).
- (2) Atomic flows provide for an efficient way to control substitutions for atom occurrences in derivations. This helps us to define the cut-free form of proofs (Definition 23).
- (3) We can define graph rewriting systems over atomic flows that control normalisation procedures on derivations. This can be used to control a further refinement of the normalisation procedure (Theorem 27).

Our aim now is to quickly and informally provide the necessary notions about atomic flows, especially concerning aspects (1) and (2) above. Although the feature (3) of atomic flows, namely graph rewriting systems of flows, did help us in obtaining proofs in normal form, we estimate that formally introducing the necessary machinery is unjustified in this paper. In fact, given our limited needs here, we can operate directly on derivations, without the intermediate support of atomic flows. Nonetheless, being aware of the underlying atomic-flow methods is useful for the reader who wishes to further investigate this matter. So, we informally provide, in Section 7, enough material to make the connection with the atomic-flow techniques that are fully developed in [GG08].

We obtain one atomic flow from each derivation by tracing all its atom occurrences and by keeping track of their creation and destruction (in identity/cut and weakening/co-weakening instances), their duplication (in contraction/cocontraction) and their duality (in identity/cut). Technically, atomic flows are directed graphs of a special kind, but it is more intuitive to consider them as diagrams generated by composing *elementary atomic flows* that belong to one of seven kinds.

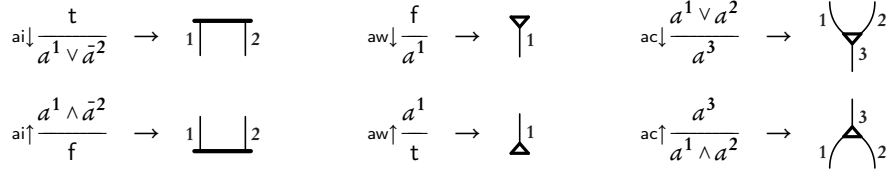


FIGURE 1. Vertices of atomic flows.

The first kind of elementary atomic flow is the *edge*



which corresponds to one or more occurrences of the same atom in a given derivation, all of which are not active in any structural rule instance, *i.e.*, they are not the atom occurrences that instantiate a structural rule.

The other six kinds of elementary diagrams are associated with the six structural inference rules, as shown in Figure 1, and they are called *vertices*; each vertex has some incident edges. At the left of each arrow, we see an instance of a structural rule, where the atom occurrences are labelled by small numerals; at the right of the arrow, we see the vertex corresponding to the rule instance, whose incident edges are labelled in accord with the atom occurrences they correspond to. We qualify each vertex according to the rule it corresponds to; for example, in a given atomic flow, we might talk about a *contraction vertex*, or a *cut vertex*, and so on. Instead of small numerals, sometimes we use ϵ or ι or colour to label edges (as well as atom occurrences), but we do not always use labels.

All edges are directed, but we do not explicitly show the orientation. Instead, we consider it as implicitly given by the way we draw them, namely, edges are oriented along the vertical direction. So, the vertices corresponding to dual rules, in Figure 1, are distinct, for example, an identity vertex and a cut vertex are different because the orientation of their edges is different. On the other hand, the horizontal direction plays no role in distinguishing atomic flows; this corresponds to commutativity of logical relations.

We can define (*atomic*) *flows* as the smallest set of diagrams containing elementary atomic flows, and closed under the composition operation consisting in identifying zero or more edges such that no cycle is created. In addition, for a diagram to be an atomic flow, it must be possible to assign it a polarity, according to the following definition. A *polarity assignment* is a mapping of each edge to an element of $\{-, +\}$, such that the two edges of each identity or cut vertex map to different values and the three edges of each contraction or cocontraction vertex map to the same value. We denote atomic flows by ϕ and ψ .

Let us see some examples. The flow

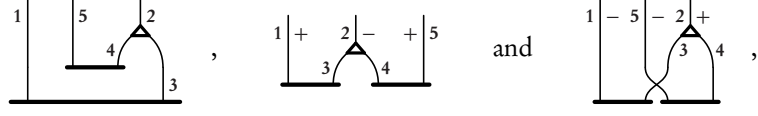


is obtained by juxtaposing (*i.e.*, composing by identifying zero edges):

- three edges,
- a flow obtained by composing a cut vertex with a cocontraction vertex, and
- a flow obtained by composing an identity vertex with a cut vertex.

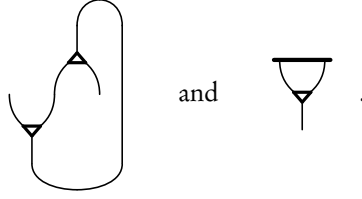
Note that there are no cycles in the flow, and that we can find 32 different polarity assignments, *i.e.*, two for each of the five connected components of the flow (this is a general rule).

Let us see another example. These are three different representations of the same flow:



where we label edges to show their correspondence. In the two rightmost flows, we indicate the two different polarity assignments that are possible.

The following two diagrams are not atomic flows:



The left one is not a flow because it contains a cycle, and the right one because there is no possible polarity assignment.

Let us see how to extract atomic flows from derivations. Given an SKS derivation Φ , we obtain, by the following prescriptions, a unique atomic flow ϕ , such that there is a surjective map between atom occurrences in Φ and edges of ϕ :

- Each structural inference step in Φ is associated with one and only one vertex in ϕ , such that active atom occurrences in the rule instance map to edges incident with the vertex. The correspondence is indicated in Figure 1. For example, the flow associated with the inference step at the left is indicated at the right:

$$\text{ac}\downarrow \frac{a^1 \wedge [b^2 \vee [a^3 \vee a^4]]}{a^1 \wedge [b^2 \vee a^5]} \quad \text{and} \quad \begin{array}{c} 1 \\ | \\ 2 \\ | \\ 3 \vee 4 \\ \vee \\ 5 \end{array} .$$

Note that the nonactive atoms are ‘traced’ by associating each trace with one edge; this corresponds well to abbreviating, say, the inference step $\text{ac}\downarrow \frac{K\{a \vee a\}}{K\{a\}}$ by $K\left\{\frac{a \vee a}{a}\right\}$.

- For each other inference step in Φ , all the atom occurrences in the premiss are respectively mapped to the same edges of ϕ as the atom occurrences in the conclusion. For example, the flow associated with the inference step

$$\text{m} \frac{a^1 \wedge [(b^2 \wedge c^3) \vee (d^4 \wedge e^5)]}{a^1 \wedge ([b^2 \vee d^4]) \wedge ([c^3 \vee e^5])} \quad \text{is} \quad \begin{array}{c} 1 \\ | \\ 2 \\ | \\ 3 \\ | \\ 4 \\ | \\ 5 \end{array} .$$

The flow ϕ so obtained is called the atomic flow *associated with* the derivation Φ . We show three examples in Figure 2: in the top row we see three SKS derivations in the standard CoS syntax; in the row below, we show the same derivations in the open deduction notation; in the bottom row, we see the three corresponding atomic flows.

Perhaps surprisingly, it can be proved that every flow is associated with infinitely many SKS derivations (see [GG08]).

We introduce now some graphical shortcuts. When certain details of a flow are not important, but only the vertex kinds and its upper and lower edges are, we can use boxes, labelled with all the vertex kinds that can appear in the flow they represent. For example, the following left and centre flows could represent the previously seen flow (2), whereas

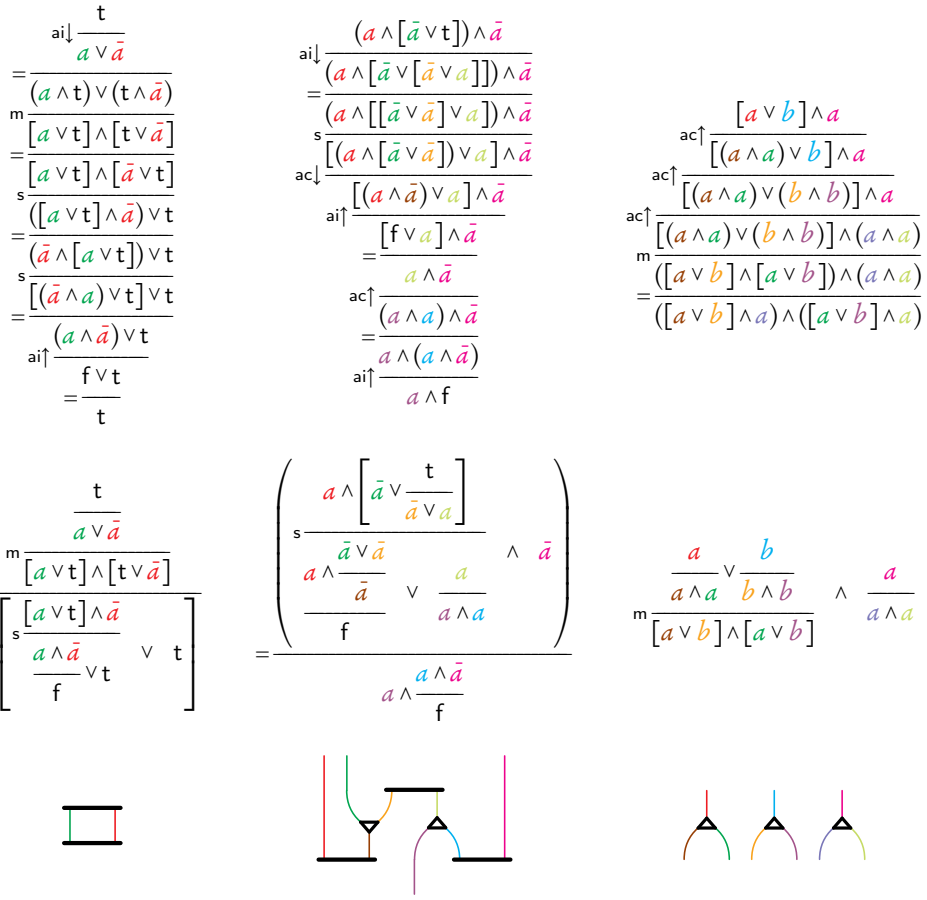
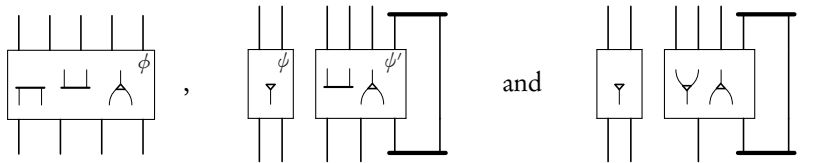


FIGURE 2. Examples of derivations in CoS and open deduction notation, and associated atomic flows.

the right flow cannot:



The flow at the right cannot represent flow (2) because it has the wrong number of lower edges and because a necessary cut vertex is not allowed by the labelling of the boxes. As just shown, we sometimes label boxes with the name of the flow they represent. For example, flow ϕ above could represent flow (2), and, if the centre flow stands for (2), then flows ψ and ψ' are, respectively,



When no vertex labels appear on a box, we assume that the vertices in the corresponding flow can be any (so, it does not mean that there are no vertices in the flow).

We can then distinguish between the three occurrences of \bar{a} that are mapped to edge 1 and the one that is not, as in

$$\Phi = \left[\begin{array}{c} (a \wedge f) \vee \left(a \wedge \frac{f}{\bar{a}^1} \right) \\ \frac{a \vee a}{a} \wedge \frac{f \vee \bar{a}^1}{\bar{a}^1} \\ \frac{\quad}{f} \end{array} \vee \bar{a} \right] ;$$

we can also substitute for these occurrences, for example by $\{\bar{a}^1/f\}$; such a situation occurs in the proof of Theorem 11. Note that simply substituting f for \bar{a}^1 would invalidate this derivation because it would break the cut and weakening instances; however, the proof of Theorem 11 specifies how to fix such broken instances.

We generalise this labelling mechanism to boxes. For example, we can use a different representation of the flow of Φ to individuate two classes a^ϕ and \bar{a}^ϕ of atom occurrences, as follows:

$$\Phi = \left[\begin{array}{c} (a \wedge f) \vee \left(a \wedge \frac{f}{\bar{a}^\phi} \right) \\ \frac{a \vee a}{a^\phi} \wedge \frac{f \vee \bar{a}^\phi}{\bar{a}^\phi} \\ \frac{\quad}{f} \end{array} \right] \vee \bar{a}^\phi \quad \text{and} \quad \begin{array}{c} \text{Y} \\ | \\ \text{Box } \phi \\ | \\ \text{Y} \end{array} .$$

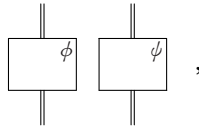
In order to define the notion of cut-free form (Definition 23), we need the following proposition, which we state here because it constitutes a good exercise about atomic flows. Note that, in the following, we use several boxes labelled by ϕ : this means that we are dealing with several copies of the same flow ϕ .

Notation 2. Given a formula A in a derivation whose associated atomic flow contains a flow ϕ , we indicate with a^ϕ every occurrence of the atom a in A whose associated edge is in ϕ . So, as in the following Proposition 3, $A\{a^\phi/B, \bar{a}^\phi/\bar{B}\}$ stands for the formula A where the atom occurrences of a and its dual, whose associated edges are in ϕ , are substituted with formula B and its dual, respectively.

Proposition 3. *Given a derivation*

$$\frac{A}{\Phi \parallel_{\text{SKS}} A'}$$

let its associated flow have shape



such that ϕ is a connected component each of whose edges is associated with atom a or \bar{a} ; then, for any formula B , there exists a derivation

$$\frac{A\{a^\phi/B, \bar{a}^\phi/\bar{B}\}}{\Psi \parallel_{\text{SKS}} A'\{a^\phi/B, \bar{a}^\phi/\bar{B}\}}$$

this would invalidate identity and cut instances, but we actually only need the simple core of the proof.

Our normalisation procedure essentially relies on gluing together simple cores, where we substitute the a_i atom occurrences that map to edges in ϕ_i with certain formulae called ‘pseudocomplements’ (see Section 5 and Definition 23).

Remark 8. A proof in simple form over \mathfrak{a}_1^0 is cut-free.

In order to prove Theorem 11, we need two facts, Proposition 9 and Lemma 10.

In the following (routine) proposition, we use the switch rule s to ‘push outside’ or ‘pull inside’ a formula A , relative to a context $K\{ \}$.

Proposition 9. *For any context $K\{ \}$ and formula A , there exist derivations whose size is less than $|K\{A\}|^2$ and have shape*

$$\begin{array}{c} K\{A\} \\ \parallel_{\{s\}} \\ A \vee K\{f\} \end{array} \quad \text{and} \quad \begin{array}{c} A \wedge K\{t\} \\ \parallel_{\{s\}} \\ K\{A\} \end{array} .$$

Proof. We only build the derivation at the left in the claim, the construction being dual for the one at the right. We reason by induction on the number n of $\vee\text{-}\wedge$ alternations in the formula-tree branch of $\{ \}$ in $K\{ \}$. If $n = 0$, then $K\{A\} = A \vee K\{f\}$. If $n > 0$, consider

$$\left[\begin{array}{c} \left(\begin{array}{c} H\{A\} \\ \parallel_{\{s\}} \\ A \vee H\{f\} \end{array} \wedge B \right) \vee C \\ \underset{s}{\parallel} \\ A \vee (H\{f\} \wedge B) \end{array} \right] ,$$

for some context $H\{ \}$ and formulae B and C , such that $K\{ \} = (H\{ \} \wedge B) \vee C$ and the number of $\vee\text{-}\wedge$ alternations in the formula-tree branch of $\{ \}$ in $H\{ \}$ is $n-1$. The number of s instances is n , and we have that $n \leq |K\{f\}|$. \square

Note that the atomic flows of the derivations in the previous proposition only consist of edges because no structural rules appear.

To prove Theorem 11, we could now proceed as follows. Given a proof, we assign it (and its flow) an arbitrary polarity, under certain assumptions that we can always easily satisfy. We then focus on the negative paths connecting identity and cut vertices. If cocontraction vertices lie along these paths, we have a potential problem because some atoms in the conclusion of the proof might be connected to atoms in some identity instances. This would prevent us from substituting pseudocomplements, as previously mentioned, because by doing so we would alter the conclusion of the proof.

However, we can solve the problem by replacing each cocontraction vertex by an appropriate flow involving identity, cut and contraction vertices, in such a way that the only contraction vertex so introduced is positive. Actually, the lemma below takes a more radical approach, which simplifies exposition and also has broader application: we replace all negative contraction and cocontraction instances. This unnecessarily bloats the proof, but still stays well inside polynomial bounds.

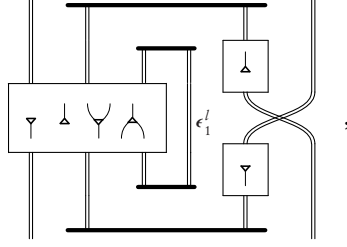
Lemma 10. *Given any derivation*

$$\begin{array}{c} A \\ \Phi \parallel_{\text{SKS}} \\ B \end{array} ,$$

we can, in linear time in the size of Φ , construct a derivation

$$\begin{array}{c} A \\ \parallel_{\text{SKS}} \\ B \end{array}$$

such that its atomic flow has shape

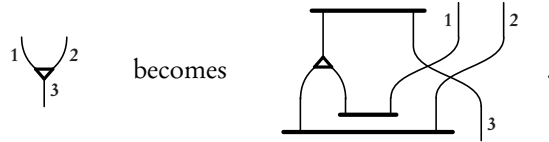


and such that no two atoms associated with $\epsilon_1, \dots, \epsilon_l$ are dual, for some $l \geq 0$.

Proof. Assign a polarity to the flow of Φ such that no two dual atoms are both associated with negative edges; then replace each negative contraction instance as follows:

$$\begin{array}{c} A \\ \Psi \parallel_{\text{SKS}} \\ K \left\{ \begin{array}{c} \bar{a} \vee \bar{a} \\ \bar{a} \end{array} \right\} \\ \Psi' \parallel_{\text{SKS}} \\ B \end{array} \quad \text{becomes} \quad K \left\{ \begin{array}{c} \frac{t}{a \vee \bar{a}} \wedge [\bar{a} \vee \bar{a}] \\ \frac{a}{a \wedge a} \wedge [\bar{a} \vee \bar{a}] \\ \frac{s}{a \wedge [(a \wedge \bar{a}) \vee \bar{a}]} \vee \bar{a} \\ \frac{s}{\frac{a \wedge \bar{a}}{f} \vee \frac{a \wedge \bar{a}}{f}} \\ \Psi' \parallel_{\text{SKS}} \\ B \end{array} \right\} .$$

This corresponds, in the flow, to replacing each negative contraction vertex as follows:



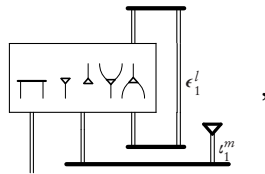
Proceed analogously with negative cocontraction instances. \square

We are now ready to prove the main result of this section.

Theorem 11. *Given any proof Π of A in SKS, we can, in cubic time in the size of Π , construct a proof of A in simple form.*

Proof. We proceed in three steps.

- (1) By Lemma 10, we can transform Π , in linear time in its size, into a proof $\Pi' \parallel_{\text{SKS}}^A$, whose flow has shape

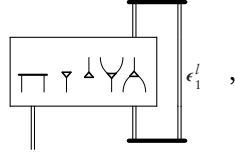


where $l, m \geq 0$ and such that no two atoms associated with $\epsilon_1, \dots, \epsilon_l$ are dual. For $1 \leq i \leq m$, we successively transform Π' as follows, for some $\Pi'', \Phi, \Phi', K \{ \}$

and $H\{ \}$ }:

$$\begin{array}{c} \Pi'' \parallel \\ K \left\{ \frac{f}{\bar{a}^{l_i}} \right\} \\ \Phi \parallel \\ H \left\{ \frac{a \wedge \bar{a}^{l_i}}{f} \right\} \\ \Phi' \parallel \\ A \end{array} \quad \text{becomes} \quad \begin{array}{c} \Pi'' \parallel \\ K \{f\} \\ \Phi \{ \bar{a}^{l_i} / f \} \parallel \\ H \left\{ \frac{a}{t} \wedge f \right\} \\ \Phi' \parallel \\ A \end{array} .$$

This way, we obtain, in linear time, a proof $\Pi''' \parallel_A^{\text{SKS}}$, whose flow is



and whose size is smaller than $|\Pi'|$.

- (2) Thanks to Proposition 9, for $1 \leq i \leq l$, we successively transform Π''' as follows, for some Ψ, Ψ' and $K'\{ \}$ }:

$$\begin{array}{c} \Psi \parallel \\ K' \left\{ \frac{t}{a \vee \bar{a}^{\epsilon_i}} \right\} \\ \Psi' \parallel \\ A \end{array} \quad \text{becomes} \quad \begin{array}{c} t \\ a \vee \bar{a}^{\epsilon_i} \\ [a \vee \bar{a}] \wedge \Psi \parallel \\ [a \vee \bar{a}^{\epsilon_i}] \wedge K' \{t\} \\ \parallel \{s\} \\ K' \{a \vee \bar{a}^{\epsilon_i}\} \\ \Psi' \parallel \\ A \end{array} ;$$

we also apply the dual transformation for each $a_i \uparrow$ instance. This way, we obtain a proof

$$\begin{array}{c} \frac{t}{a_1 \vee \bar{a}_1^{\epsilon_1}} \wedge \dots \wedge \frac{t}{a_l \vee \bar{a}_l^{\epsilon_l}} \\ \Psi'' \parallel \\ A \vee \frac{a_1 \wedge \bar{a}_1^{\epsilon_1}}{f} \vee \dots \vee \frac{a_l \wedge \bar{a}_l^{\epsilon_l}}{f} \end{array} ,$$

whose flow is the same as that of Π''' because each transformation conserves the flow. If $|\Pi'''| = n$, and given that $n > 2l$, the size of each derivation introduced by virtue of Proposition 9 is at most $4n^2$. So, each of the $2l$ transformations increases the size of the proof by $O(n^2)$, which makes for a total complexity of $O(n^3)$.

- (3) Consider \mathbf{b}_1^n such that b_1, \dots, b_n are distinct and $\{a_1, \dots, a_l\} = \{b_1, \dots, b_n\}$. We can build, in linear time, the proof

$$\left[A \vee \begin{array}{c} \frac{t}{b_1 \vee \bar{b}_1} \wedge \dots \wedge \frac{t}{b_n \vee \bar{b}_n} \\ \parallel \{\uparrow\} \\ [a_1 \vee \bar{a}_1^{\epsilon_1}] \wedge \dots \wedge [a_l \vee \bar{a}_l^{\epsilon_l}] \\ \parallel \Psi'' \\ (a_1 \wedge \bar{a}_1^{\epsilon_1}) \vee \dots \vee (a_l \wedge \bar{a}_l^{\epsilon_l}) \\ \parallel \{\downarrow\} \\ \left[\frac{b_1 \wedge \bar{b}_1}{f} \vee \dots \vee \frac{b_n \wedge \bar{b}_n}{f} \right] \end{array} \right],$$

which is in simple form over \mathbf{b}_1^n . We can then obtain a proof in SKS in time $O(n^2)$, because of Proposition 1. \square

The transformation in Step (1) in the previous proof is a case of ‘weakening reduction’ for atomic flows, studied in [GG08]. In Section 7 we comment more on this.

Remark 12. In general, given a proof Π and by the construction in the proof of Theorem 11, we can obtain several different simple forms from Π . In fact, apart from permutations of rule instances, commutativity and associativity, the simple forms depend on the choice of a polarity assignment (Lemma 10).

5. THRESHOLD FORMULAE

We present here the main construction of this paper, *i.e.*, a class of derivations Γ that only depend on a given set of atoms and that allow us to normalise any proof containing those atoms. The complexity of the Γ derivations dominates the complexity of the normal proof, and is due to the complexity of certain ‘threshold formulae’, on which the Γ derivations are based. The Γ derivations are constructed in Definition 19; this directly leads to Theorem 21, which states a crucial property of the Γ derivations and which is the main result of this section.

Threshold formulae realise boolean threshold functions, which are defined as boolean functions that are true if and only if at least k of n inputs are true (see [Weg87] for a thorough reference on threshold functions).

In the following, $\lfloor x \rfloor$ denotes the maximum integer n such that $n \leq x$.

There are several ways of encoding threshold functions into formulae, and the problem is to find, among them, an encoding that allows us to obtain Theorem 21. Efficiently obtaining the property stated in Theorem 21 crucially depends also on the proof system we adopt.

The following class of threshold formulae, which we found to work for system SKS, is a simplification of the one adopted in [AGP02].

Definition 13. Consider $n > 0$, distinct atoms a_1, \dots, a_n , and let $p = \lfloor n/2 \rfloor$ and $q = n - p$; for $k \geq 0$, we define the *threshold formulae* $\theta_k^n \mathbf{a}_1^n$ as follows:

- for any $n > 0$ let $\theta_0^n \mathbf{a}_1^n \equiv t$;
- for any $n > 0$ and $k > n$ let $\theta_k^n \mathbf{a}_1^n \equiv f$;
- $\theta_1^1(a_1) \equiv a_1$;
- for any $n > 1$ and $0 < k \leq n$ let $\theta_k^n \mathbf{a}_1^n \equiv \bigvee_{\substack{i+j=k \\ 0 \leq i \leq p \\ 0 \leq j \leq q}} (\theta_i^p \mathbf{a}_1^p \wedge \theta_j^q \mathbf{a}_{p+1}^q)$.

$$\begin{aligned}
\theta_0^2(a, b) &\equiv \mathbf{t} \quad , \\
\theta_1^2(a, b) &\equiv (\theta_1^1(a) \wedge \theta_0^1(b)) \vee (\theta_0^1(a) \wedge \theta_1^1(b)) \equiv (a \wedge \mathbf{t}) \vee (\mathbf{t} \wedge b) \\
&= a \vee b \quad , \\
\theta_2^2(a, b) &\equiv \theta_1^1(a) \wedge \theta_1^1(b) \\
&= a \wedge b \quad , \\
\theta_0^3(a, b, c) &\equiv \mathbf{t} \quad , \\
\theta_1^3(a, b, c) &\equiv (\theta_1^1(a) \wedge \theta_0^2(b, c)) \vee (\theta_0^1(a) \wedge \theta_1^2(b, c)) \equiv (a \wedge \mathbf{t}) \vee (\mathbf{t} \wedge [(b \wedge \mathbf{t}) \vee (\mathbf{t} \wedge c)]) \\
&= a \vee b \vee c \quad , \\
\theta_2^3(a, b, c) &\equiv (\theta_1^1(a) \wedge \theta_1^1(b, c)) \vee (\theta_0^1(a) \wedge \theta_2^2(b, c)) \\
&= (a \wedge [b \vee c]) \vee (b \wedge c) \quad , \\
\theta_3^3(a, b, c) &\equiv \theta_1^1(a) \wedge \theta_2^2(b, c) \equiv (a \wedge (b \wedge c)) \\
&= a \wedge b \wedge c \quad , \\
\theta_0^5(a, b, c, d, e) &\equiv \mathbf{t} \quad , \\
\theta_1^5(a, b, c, d, e) &\equiv (\theta_1^2(a, b) \wedge \theta_0^3(c, d, e)) \vee (\theta_0^2(a, b) \wedge \theta_1^3(c, d, e)) \\
&= a \vee b \vee c \vee d \vee e \quad , \\
\theta_2^5(a, b, c, d, e) &\equiv (\theta_2^2(a, b) \wedge \theta_0^3(c, d, e)) \vee (\theta_1^2(a, b) \wedge \theta_1^3(c, d, e)) \vee (\theta_0^2(a, b) \wedge \theta_2^3(c, d, e)) \\
&= (a \wedge b) \vee ([a \vee b] \wedge [c \vee d \vee e]) \vee (c \wedge [d \vee e]) \vee (d \wedge e) \quad , \\
\theta_3^5(a, b, c, d, e) &\equiv (\theta_2^2(a, b) \wedge \theta_1^3(c, d, e)) \vee (\theta_1^2(a, b) \wedge \theta_2^3(c, d, e)) \vee (\theta_0^2(a, b) \wedge \theta_3^3(c, d, e)) \\
&= (a \wedge b \wedge [c \vee d \vee e]) \vee ([a \vee b] \wedge [(c \wedge [d \vee e]) \vee (d \wedge e)]) \vee (c \wedge d \wedge e) \quad , \\
\theta_4^5(a, b, c, d, e) &\equiv (\theta_2^2(a, b) \wedge \theta_2^3(c, d, e)) \vee (\theta_1^2(a, b) \wedge \theta_3^3(c, d, e)) \\
&= (a \wedge b \wedge [(c \wedge [d \vee e]) \vee (d \wedge e)]) \vee ([a \vee b] \wedge c \wedge d \wedge e) \quad , \\
\theta_5^5(a, b, c, d, e) &\equiv \theta_2^2(a, b) \wedge \theta_3^3(c, d, e) \\
&= a \wedge b \wedge c \wedge d \wedge e \quad , \\
\theta_6^5(a, b, c, d, e) &\equiv \mathbf{f} \quad .
\end{aligned}$$

FIGURE 3. Examples of threshold formulae.

Compared to the definition in [AGP02], we require $i + j = k$ instead of $i + j \geq k$. The semantics of the formulae does not change but their size is smaller and their structure is much simpler, arguably benefiting further research. See, in Figure 3, some examples of threshold formulae.

The only reason why we require atoms to be distinct in threshold formulae is to avoid certain technical problems with substitutions in the definition of cut-free form, later on. However, there is no substantial difficulty in relaxing this definition to any set of atoms.

The formulae for threshold functions adopted in [AGP02] correspond, for each choice of k and n , to $\bigvee_{i \geq k} \theta_i^n \mathbf{a}_1^n$. We presume that [AGP02] employs these more complicated formulae because the formalism adopted there, the sequent calculus, is less flexible than deep inference, requiring more information in threshold formulae in order to construct suitable derivations.

Remark 14. For $n > 0$, we have $\theta_1^n \mathbf{a}_1^n = a_1 \vee \dots \vee a_n$ and $\theta_n^n \mathbf{a}_1^n = a_1 \wedge \dots \wedge a_n$.

The size of the threshold formulae dominates the cost of the normalisation procedure, so, we evaluate their size. We leave as an exercise the proof of the following proposition.

Proposition 15. *For any $n > 0$ and $k \geq 0$, $|\theta_k^n \mathbf{a}_1^n| \leq |\theta_{\lfloor n/2 \rfloor + 1}^n \mathbf{a}_1^n|$.*

Lemma 16. *The size of $\theta_{\lfloor n/2 \rfloor + 1}^n \mathbf{a}_1^n$ is $n^{O(\log n)}$.*

Proof. Observe that $|\theta_k^n \mathbf{a}_1^n| \leq |\theta_k^{n+1} \mathbf{a}_1^{n+1}|$. Let $p = \lfloor n/2 \rfloor$ and $q = n - p$ and consider:

$$\begin{aligned}
 (3) \quad |\theta_{p+1}^n \mathbf{a}_1^n| &= \sum_{\substack{i+j=p+1 \\ 0 \leq i \leq p \\ 0 \leq j \leq q}} \left(|\theta_i^p \mathbf{a}_1^p| + |\theta_j^q \mathbf{a}_{p+1}^q| \right) \\
 &\leq \sum_{\substack{i+j=p+1 \\ 0 \leq i, j \leq q}} \left(|\theta_i^q \mathbf{a}_1^q| + |\theta_j^q \mathbf{a}_1^q| \right) \\
 &\leq 2(q+1) |\theta_{\lfloor q/2 \rfloor + 1}^q \mathbf{a}_1^q| \quad ,
 \end{aligned}$$

where we use Proposition 15. We show that, for $h = 2/(\log 3 - \log 2)$ and for any $n > 0$, we have $|\theta_{\lfloor n/2 \rfloor + 1}^n \mathbf{a}_1^n| \leq n^{h \log n}$. We reason by induction on n ; the case $n = 1$ trivially holds. By the inequality (3), and for $n > 1$, we have

$$\begin{aligned}
 |\theta_{\lfloor n/2 \rfloor + 1}^n \mathbf{a}_1^n| &\leq 2(n - \lfloor n/2 \rfloor + 1)(n - \lfloor n/2 \rfloor)^{h \log(n - \lfloor n/2 \rfloor)} \\
 &\leq n^2 n^{h \log(2n/3)} = n^{h \log n - h(\log 3 - \log 2) + 2} = n^{h \log n} \quad . \quad \square
 \end{aligned}$$

Theorem 17. *For any $k \geq 0$ the size of $\theta_k^n \mathbf{a}_1^n$ is $n^{O(\log n)}$.*

Proof. It immediately follows from Proposition 15 and Lemma 16. □

Given a threshold formula $\theta_k^n \mathbf{a}_1^n$, we can consider, for each a_l such that $1 \leq l \leq n$, the formulae $(\theta_k^n \mathbf{a}_1^n)\{a_l/f\}$ and $(\theta_{k+1}^n \mathbf{a}_1^n)\{a_l/t\}$: we call both of them, informally, ‘pseudocomplements’ of a_l . The reason for this name is that we can manage to replace, in a given proof, all occurrences of those \bar{a}_l that appear in cut instances with the pseudocomplements of a_l . The cut instances and their corresponding identity instances are then removed, leaving us with derivations whose premiss and conclusion contain each a threshold formula. Moreover, the k -level of the threshold formula in the premiss is one less than the k -level of the threshold formula in the conclusion. This way, we obtain several derivations, corresponding to increasing values of k , that we are able to stitch together until we get a normalised proof.

All this, of course, needs clarification and for many it might be helpful only after having grasped the full proof in its technical form. However, we think that it is convenient here to provide a summary of the main constructions allowing for this stitching operation. Let us read derivations top-down; the following are the steps that we need to perform, for $0 \leq k \leq n$.

(1) Build

$$\begin{array}{c}
 \theta_k^n \mathbf{a}_1^n \\
 \parallel \\
 a_l \vee (\theta_k^n \mathbf{a}_1^n)\{a_l/f\}
 \end{array} \quad ,$$

i.e., create, from a k -level threshold formula, a disjunction between a_l and its pseudocomplement $(\theta_k^n \mathbf{a}_1^n)\{a_l/f\}$ (Proposition 22); then replace the pseudocomplement into \bar{a}_l , for each identity instance.

(2) Increase the k -level by using the derivations

$$\begin{array}{c}
 (\theta_k^n \mathbf{a}_1^n)\{a_l/f\} \\
 \parallel \\
 (\theta_{k+1}^n \mathbf{a}_1^n)\{a_l/t\}
 \end{array}$$

(Theorem 21); these are the Γ derivations mentioned in the introduction to this section.

- (3) For each cut instance, collect the conjunction between a_l and its pseudocomplement $(\theta_{k+1}^n a_1^n)\{a_l/t\}$; then build

$$a_l \wedge (\theta_{k+1}^n a_1^n)\{a_l/t\} \parallel \theta_{k+1}^n a_1^n ,$$

i.e., create a $(k+1)$ -level threshold formula (Proposition 22).

The derivations mentioned above do not require any use of identity and cut, and allow us to move, in $n+1$ steps, from $\theta_0^n a_1^n \equiv t$ to $\theta_{n+1}^n a_1^n \equiv f$, which is the secret to success. The constructions in 1 and 3 are deep-inference routine and introduce low complexity. We deal now with the crucial step 2, by designing Definition 19, and then checking it carefully, so as to get the property stated in Theorem 21.

Definition 19 is technical, but its philosophy is simple; all one has to do to build the derivations required by Theorem 21 is:

- identify the atom occurrences that must occur in the premiss and that must not occur in the conclusion and remove them using coweakening, and
- identify the atom occurrences that must occur in the conclusion and that must not occur in the premiss and add them using weakening.

We have implemented Definition 19 as a program [Gug09]. It can be useful to read the definition together with the examples in Figures 4 and 3, which have been generated by the program.

Remark 18. Given $n > 1$, let $p = \lfloor n/2 \rfloor$ and $q = n - p$. For $0 \leq k \leq q$ and $1 \leq l \leq p$, the following derivation is well defined:

$$w\uparrow \frac{(\theta_p^p a_1^p)\{a_l/f\} \wedge \theta_k^q a_{p+1}^n}{f} = w\uparrow \frac{a_1 \wedge \dots \wedge a_{l-1} \wedge a_{l+1} \wedge \dots \wedge a_p \wedge \theta_k^q a_{p+1}^n}{t} \wedge f .$$

Analogously, for $0 \leq k \leq p$ and $p+1 \leq l \leq n$, we can define the following derivation:

$$w\uparrow \frac{\theta_k^p a_1^p \wedge (\theta_q^q a_{p+1}^n)\{a_l/f\}}{f} = w\uparrow \frac{\theta_k^p a_1^p \wedge a_{p+1} \wedge \dots \wedge a_{l-1} \wedge a_{l+1} \wedge \dots \wedge a_n}{t} \wedge f .$$

Both classes of derivations are used in Definition 19.

Definition 19. Consider $n > 0$, distinct atoms a_1, \dots, a_n , and let $p = \lfloor n/2 \rfloor$ and $q = n - p$.

- For $n > 1$ and $1 \leq l \leq n$, we define the derivations $\Upsilon_{k,l}^n a_1^n$ and $\Delta_{k,l}^n a_1^n$ as follows:

$$\Upsilon_{k,l}^n a_1^n = \begin{cases} w\uparrow \frac{(\theta_p^p a_1^p)\{a_l/f\} \wedge \theta_{k-p}^q a_{p+1}^n}{f} & \text{if } p \leq k \leq n \text{ and } l \leq p \\ w\uparrow \frac{\theta_{k-q}^p a_1^p \wedge (\theta_q^q a_{p+1}^n)\{a_l/f\}}{f} & \text{if } q \leq k \leq n \text{ and } p < l \\ f & \text{otherwise} \end{cases}$$

and

$$\Delta_{k,l}^n a_1^n = \begin{cases} w\downarrow \frac{f}{\theta_k^q a_{p+1}^n} & \text{if } 0 < k \leq q \text{ and } l \leq p \\ w\downarrow \frac{f}{\theta_k^p a_1^p} & \text{if } 0 < k \leq p \text{ and } p < l \\ f & \text{otherwise} \end{cases} .$$

$$\begin{aligned}
\Gamma_{0,1}^5 \mathbf{a} &= t \vee \frac{f}{b} \vee \frac{f}{c \vee d \vee e} \quad , \\
\Gamma_{1,1}^5 \mathbf{a} &= b \vee \left(\left[t \vee \frac{f}{b} \right] \wedge [c \vee d \vee e] \right) \vee \frac{f}{(c \wedge [d \vee e]) \vee (d \wedge e)} \quad , \\
\Gamma_{2,1}^5 \mathbf{a} &= (b \wedge [c \vee d \vee e]) \vee \left(\left[t \vee \frac{f}{b} \right] \wedge [(c \wedge [d \vee e]) \vee (d \wedge e)] \right) \vee \frac{f \wedge b}{f} \vee \frac{f}{c \wedge d \wedge e} \quad , \\
\Gamma_{3,1}^5 \mathbf{a} &= (b \wedge [(c \wedge [d \vee e]) \vee (d \wedge e)]) \vee \left(\left[t \vee \frac{f}{b} \right] \wedge c \wedge d \wedge e \right) \vee \frac{f \wedge b \wedge [c \vee d \vee e]}{f} \quad , \\
\Gamma_{4,1}^5 \mathbf{a} &= (b \wedge c \wedge d \wedge e) \vee \frac{f \wedge b \wedge [(c \wedge [d \vee e]) \vee (d \wedge e)]}{f} \quad , \\
\Gamma_{5,1}^5 \mathbf{a} &= \frac{f \wedge b \wedge c \wedge d \wedge e}{f} \quad , \\
\Gamma_{0,3}^5 \mathbf{a} &= t \vee \frac{f}{d \vee e} \vee \frac{f}{a \vee b} \quad , \\
\Gamma_{1,3}^5 \mathbf{a} &= \left([a \vee b] \wedge \left[t \vee \frac{f}{d \vee e} \right] \right) \vee d \vee e \vee \frac{f}{d \wedge e} \vee \frac{f}{a \wedge b} \quad , \\
\Gamma_{2,3}^5 \mathbf{a} &= \left(a \wedge b \wedge \left[t \vee \frac{f}{d \vee e} \right] \right) \vee \left([a \vee b] \wedge \left[d \vee e \vee \frac{f}{d \wedge e} \right] \right) \vee (d \wedge e) \vee \frac{f \wedge [d \vee e]}{f} \quad , \\
\Gamma_{3,3}^5 \mathbf{a} &= \left(a \wedge b \wedge \left[d \vee e \vee \frac{f}{d \wedge e} \right] \right) \vee \left([a \vee b] \wedge \left[(d \wedge e) \vee \frac{f \wedge [d \vee e]}{f} \right] \right) \vee \frac{f \wedge d \wedge e}{f} \quad , \\
\Gamma_{4,3}^5 \mathbf{a} &= \left(a \wedge b \wedge \left[(d \wedge e) \vee \frac{f \wedge [d \vee e]}{f} \right] \right) \vee \frac{[a \vee b] \wedge f \wedge d \wedge e}{f} \quad , \\
\Gamma_{5,3}^5 \mathbf{a} &= \frac{a \wedge b \wedge f \wedge d \wedge e}{f} \quad , \\
\Gamma_{0,5}^5 \mathbf{a} &= t \vee \frac{f}{d} \vee \frac{f}{c} \vee \frac{f}{a \vee b} \quad , \\
\Gamma_{1,5}^5 \mathbf{a} &= \left([a \vee b] \wedge \left[t \vee \frac{f}{d} \vee \frac{f}{c} \right] \right) \vee \left(c \wedge \left[t \vee \frac{f}{d} \right] \right) \vee d \vee \frac{f}{a \wedge b} \quad , \\
\Gamma_{2,5}^5 \mathbf{a} &= \left(a \wedge b \wedge \left[t \vee \frac{f}{d} \vee \frac{f}{c} \right] \right) \vee \left([a \vee b] \wedge \left[\left(c \wedge \left[t \vee \frac{f}{d} \right] \right) \vee d \right] \right) \vee (c \wedge d) \vee \frac{d \wedge f}{f} \quad , \\
\Gamma_{3,5}^5 \mathbf{a} &= \left(a \wedge b \wedge \left[\left(c \wedge \left[t \vee \frac{f}{d} \right] \right) \vee d \right] \right) \vee \left([a \vee b] \wedge \left[(c \wedge d) \vee \frac{d \wedge f}{f} \right] \right) \vee \frac{c \wedge d \wedge f}{f} \quad , \\
\Gamma_{4,5}^5 \mathbf{a} &= \left(a \wedge b \wedge \left[(c \wedge d) \vee \frac{d \wedge f}{f} \right] \right) \vee \frac{[a \vee b] \wedge c \wedge d \wedge f}{f} \quad , \\
\Gamma_{5,5}^5 \mathbf{a} &= \frac{a \wedge b \wedge c \wedge d \wedge f}{f} \quad .
\end{aligned}$$

FIGURE 4. Examples of $\Gamma_{k,l}^5 \mathbf{a}$, where $\mathbf{a} = (a, b, c, d, e)$.

- For $k \geq 0$ and $1 \leq l \leq n$, we define the derivations $\Gamma_{k,l}^n \mathbf{a}_1^n$, recursively on n , as follows:
 - $\Gamma_{0,1}^1(a_1) = t$;
 - for $k > 0$, $\Gamma_{k,1}^1(a_1) = f$;
 - for $k > n$, $\Gamma_{k,l}^n \mathbf{a}_1^n = f$;
 - for $n > 1$ and $k \leq n$, let

$$\Gamma_{k,l}^n \mathbf{a}_1^n = \begin{cases} \bigvee_{\substack{i+j=k \\ 0 \leq i < p \\ 0 \leq j \leq q}} \left(\Gamma_{i,l}^p \mathbf{a}_1^p \wedge \Theta_j^q \mathbf{a}_{p+1}^n \right) \vee \Upsilon_{k,l}^n \mathbf{a}_1^n \vee \Delta_{k+1,l}^n \mathbf{a}_1^n & \text{if } l \leq p \\ \bigvee_{\substack{i+j=k \\ 0 \leq i \leq p \\ 0 \leq j < q}} \left(\Theta_i^p \mathbf{a}_1^p \wedge \Gamma_{j,l-p}^q \mathbf{a}_{p+1}^n \right) \vee \Upsilon_{k,l}^n \mathbf{a}_1^n \vee \Delta_{k+1,l}^n \mathbf{a}_1^n & \text{if } p < l \end{cases} .$$

Example 20. See, in Figure 4, some example of derivations $\Gamma_{k,l}^n \mathbf{a}_1^n$. Note that, for clarity, we removed all instances of the trivial derivations $\Upsilon_{1,1}^2 \mathbf{a}_1^2 = \Upsilon_{1,2}^2 \mathbf{a}_1^2 = \Upsilon_{1,1}^3 \mathbf{a}_1^3 = w \uparrow \frac{f}{f}$. We can do so because these derivation instances appear as disjuncts.

Theorem 21. For any $n > 0$, $k \geq 0$ and $1 \leq l \leq n$, the derivation $\Gamma_{k,l}^n \mathbf{a}_1^n$ has shape

$$\begin{array}{c} (\Theta_k^n \mathbf{a}_1^n) \{a_l / f\} \\ \parallel \{aw \downarrow, aw \uparrow\} \\ (\Theta_{k+1}^n \mathbf{a}_1^n) \{a_l / t\} \end{array} ,$$

and $|\Gamma_{k,l}^n \mathbf{a}_1^n|$ is $n^{O(\log n)}$.

Proof. The shape of $\Gamma_{k,l}^n \mathbf{a}_1^n$ can be verified by inspecting Definition 19. For example, this is the case when $n > 1$ and $l \leq p \leq k < q$, where $p = \lfloor n/2 \rfloor$ and $q = n - p$:

$$\begin{array}{c} (\Theta_k^n \mathbf{a}_1^n) \{a_l / f\} \\ \Gamma_{k,l}^n \mathbf{a}_1^n \parallel \\ (\Theta_{k+1}^n \mathbf{a}_1^n) \{a_l / t\} \end{array} = \bigvee_{\substack{i+j=k \\ 0 \leq i < p \\ 0 \leq j \leq q}} \left(\begin{array}{c} (\Theta_i^p \mathbf{a}_1^p) \{a_l / f\} \\ \Gamma_{i,l}^p \mathbf{a}_1^p \parallel \\ (\Theta_{i+1}^p \mathbf{a}_1^p) \{a_l / t\} \end{array} \wedge \Theta_j^q \mathbf{a}_{p+1}^n \right) \vee w \uparrow \frac{(\Theta_p^p \mathbf{a}_1^p) \{a_l / f\} \wedge \Theta_{k-p}^q \mathbf{a}_{p+1}^n \vee w \downarrow \frac{f}{\Theta_{k+1}^q \mathbf{a}_{p+1}^n}}{f} .$$

(Remember that

$$\Theta_k^n \mathbf{a}_1^n \equiv \bigvee_{\substack{i+j=k \\ 0 \leq i \leq p \\ 0 \leq j \leq q}} \left(\Theta_i^p \mathbf{a}_1^p \wedge \Theta_j^q \mathbf{a}_{p+1}^n \right)$$

and $\Theta_0^p \mathbf{a}_1^p \equiv t$.) General (co)weakening rule instances can be replaced by atomic ones because of Proposition 1. The size bound on $\Gamma_{k,l}^n \mathbf{a}_1^n$ follows from Proposition 1 and Theorem 17. \square

6. NORMALISATION STEP 2: CUT-FREE FORM

In this section we define the cut-free form of proofs, based on proofs in simple form. Proofs in cut-free form have no cut instances, but can have coweakening ones, which prevent these proofs from being analytic (in the sense that atoms appear in premisses only if they do so in conclusions). Theorem 24, the main result of the section, shows how to obtain a cut-free proof from any proof. Most of the ingenuity of quasipolynomially normalising an SKS proof into one in analytic SKS resides in going from a simple form to a cut-free one. Removing coweakening instances from a cut-free form is easy; we dedicate Section 7 to this.

Before defining the cut-free form, we need to establish the following fact.

Proposition 22. *For any formula A and atom a , there exist derivations whose size is cubic in $|A|$ and that have shape*

$$\frac{A}{\|\{\text{aw}\downarrow, \text{ac}\downarrow, \text{s}\}\}} \quad \text{and} \quad \frac{a \wedge A\{a/t\}}{\|\{\text{aw}\uparrow, \text{ac}\uparrow, \text{s}\}\}} \quad .$$

Proof. If there are no occurrences of a in A , the desired derivations are

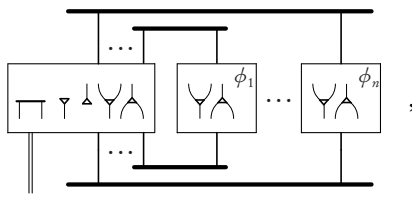
$$\frac{A}{\frac{f \vee A}{a \vee A}} \quad \text{and} \quad \frac{a \wedge A}{\frac{t \wedge A}{A}} \quad .$$

If there are $b > 0$ occurrences of a in A , obtain, by repeatedly applying Proposition 9, the following derivations:

$$\left[\frac{A}{\|\{\text{s}\}\}} \quad \text{and} \quad \left(\frac{(b-1)\text{-ac}\uparrow \frac{a}{a \wedge \dots \wedge a} \wedge A\{a/f\}}{\|\{\text{s}\}\}} \right) \right]$$

If $|A| = n$, the size of the desired derivations is $O(n^3)$ because we have to apply Proposition 9 at most $O(n)$ times. \square

Definition 23. For $n > 0$, let Π be a proof in simple form over a_1^n , such that it and its atomic flow have shape

$$\frac{\frac{t}{a_1 \vee \bar{a}_1^{\phi_1}} \wedge \dots \wedge \frac{t}{a_n \vee \bar{a}_n^{\phi_n}}}{\Psi \|\{\text{s}\}\}} \quad \text{and} \quad \text{Diagram},$$


for some derivation Ψ . For $0 \leq i \leq n+1$, let $\theta_i \equiv \theta_i^n a_1^n$. For $0 \leq k \leq n$, we define the derivations

$$\frac{\theta_k}{\Phi_k \|\{\text{SKS}\setminus\{\text{ai}\uparrow\}\}} \quad A \vee \theta_{k+1}$$

as

$$\left[\frac{(n-1)\text{-c}\uparrow \frac{\theta_k}{\left(\frac{\theta_k}{\|\{\text{aw}\downarrow, \text{ac}\downarrow, \text{s}\}\}} \wedge \dots \wedge \frac{\theta_k}{\|\{\text{aw}\downarrow, \text{ac}\downarrow, \text{s}\}\}} \right)}}{\Psi_k \|\{\text{SKS}\setminus\{\text{ai}\uparrow\}\}} \right]$$

$$\left[\frac{A \vee \left(\frac{\theta_k \{a_1/f\}}{\Gamma_{k,1}^n a_1^n \|\{\text{aw}\downarrow, \text{aw}\uparrow\}} \wedge \frac{\theta_{k+1} \{a_1/t\}}{\|\{\text{aw}\uparrow, \text{ac}\uparrow, \text{s}\}\}} \right) \vee \dots \vee \left(\frac{\theta_k \{a_n/f\}}{\Gamma_{k,n}^n a_1^n \|\{\text{aw}\downarrow, \text{aw}\uparrow\}} \wedge \frac{\theta_{k+1} \{a_n/t\}}{\|\{\text{aw}\uparrow, \text{ac}\uparrow, \text{s}\}\}} \right)}{(n-1)\text{-c}\downarrow \frac{\theta_{k+1}}{\theta_{k+1}}} \right],$$

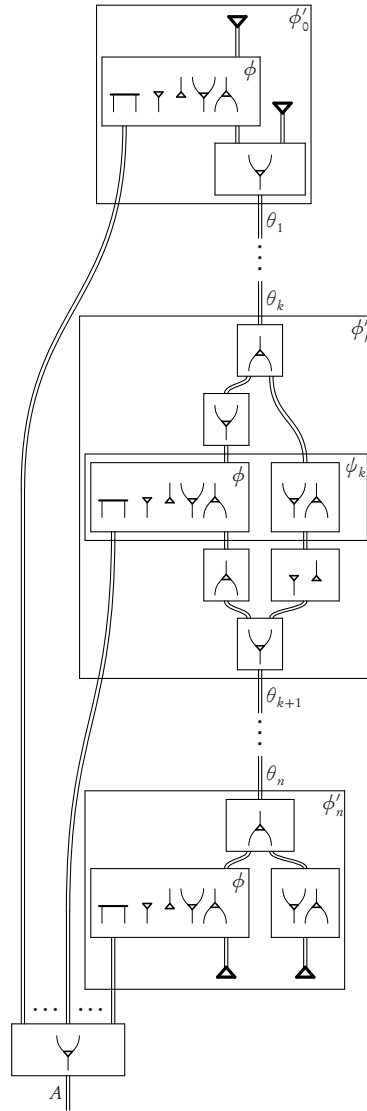
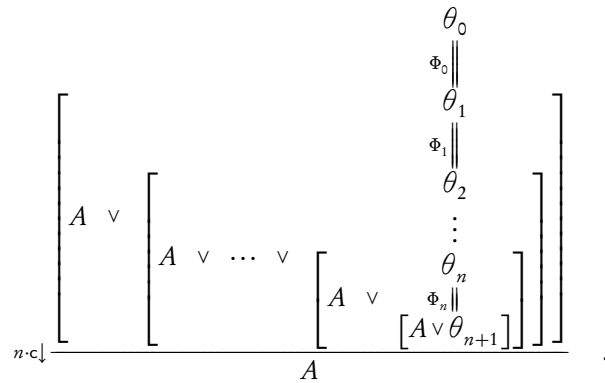


FIGURE 5. Atomic flow of a proof in cut-free form.

where $\Psi_k = \Psi\{\bar{a}_1^{\phi_1}/\theta_k\{a_1/f\}, \dots, \bar{a}_n^{\phi_n}/\theta_k\{a_n/f\}\}$ and where we use Proposition 22. We define the *cut-free form* of Π as the following proof in $\text{SKS} \setminus \{\text{ai}\}$:

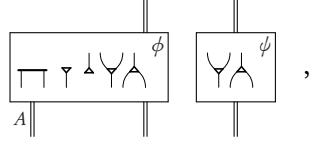


(We recall that $\theta_0 \equiv t$ and $\theta_{n+1} \equiv f$.)

Theorem 24. *Given any proof Π of A in SKS, we can construct a proof of A in $\text{SKS} \setminus \{\text{ai}\uparrow\}$ in time quasipolynomial in the size of Π .*

Proof. By Theorem 11 we can construct from Π , in polynomial time, a proof Π' of A in simple form. We can then proceed with the construction of Definition 23, to which we refer here. For $0 \leq k \leq n$, constructing Φ_k requires quasipolynomial time because of Propositions 1, 3 and 22 and Theorems 17 and 21, and because obtaining Ψ_k from Ψ requires quasipolynomial time. Constructing the cut-free form of Π' from Φ_0, \dots, Φ_n is done in polynomial time. \square

Remark 25. In Figure 5, we show the atomic flow of the cut-free form obtained from a proof Π in simple form. We refer to Definition 23. Let the following be the flow of the simple core Ψ of Π :



where ψ is the union of flows ϕ_1, \dots, ϕ_n , and where we denote by A the edges corresponding to the atom occurrences appearing in the conclusion A of Π . We then have that, for $0 < k < n$, the flow of Φ_k is ϕ'_k , as in Figure 5, where ψ_k is the flow of the derivation Ψ_k . The flows of Φ_0 and Φ_n are, respectively, ϕ'_0 and ϕ'_n .

7. NORMALISATION STEP 3: ANALYTIC FORM

Of special importance in this paper is the following proof system:

Definition 26. *Analytic SKS* is the system $\text{aSKS} = \text{SKS} \setminus \{\text{ai}\uparrow, \text{aw}\uparrow\}$.

For example, the system $\{s, m, =, \text{ac}\downarrow\}$ polynomially simulates the system $\{s, =, c\downarrow\}$, and $\text{aSKS} = \{s, m, =, \text{ai}\downarrow, \text{aw}\downarrow, \text{ac}\downarrow, \text{ac}\uparrow\}$ polynomially simulates $\{s, =, i\downarrow, w\downarrow, c\downarrow, c\uparrow\}$ (where $i\downarrow$ is the nonatomic identity). In this section, we show that we can get proofs in analytic SKS, *i.e.*, system aSKS, in quasipolynomial time from proofs in SKS.

Transforming a proof in cut-free form into an analytic one requires eliminating co-weakening rule instances. This can be done by transformations that are the dual of those over weakening instances, employed in Step (1) of the proof of Theorem 11.

Theorem 27 (Jeřábek [Jeř09]). *Given any proof Π of A in SKS, we can construct a proof of A in aSKS in time quasipolynomial in the size of Π .*

Proof. By Theorem 24, we can obtain, from Π , a cut-free proof Π' of the same formula, in quasipolynomial time in the size of Π . We associate Π' with its atomic flow ϕ , so that we have a way to identify the atom occurrences in Π' associated with each edge of ϕ , and substitute over them. We repeatedly examine each coweakening instance $\text{aw}\uparrow \frac{a^\epsilon}{t}$ in Π' , for some edge ϵ of ϕ , and we perform one transformation out of the following exhaustive list of cases, for some $\Pi'', \Phi, \Psi, K\{ \}$ and $H\{ \}$:

(1)

$$\begin{array}{c}
 \Pi'' \parallel \\
 K \left\{ \frac{t}{a^\epsilon \vee \bar{a}} \right\} \\
 \Phi \parallel \\
 H \left\{ \frac{a^\epsilon}{t} \right\} \\
 \Psi \parallel \\
 A
 \end{array}
 \text{ becomes }
 \begin{array}{c}
 \Pi'' \parallel \\
 K \left[\frac{f}{t \vee \frac{f}{\bar{a}}} \right] \\
 \Phi_{\{a^\epsilon/t\}} \parallel \\
 H\{t\} \\
 \Psi \parallel \\
 A
 \end{array}
 ;$$

(2)

$$\begin{array}{c}
 \Pi'' \parallel \\
 K \left\{ \frac{f}{a^\epsilon} \right\} \\
 \Phi \parallel \\
 H \left\{ \frac{a^\epsilon}{t} \right\} \\
 \Psi \parallel \\
 A
 \end{array}
 \text{ becomes }
 \begin{array}{c}
 \Pi'' \parallel \\
 K \left\{ \frac{f \wedge [t \vee t]}{s \frac{f \wedge t}{(f \wedge t) \vee t}} \right\} \\
 \Phi_{\{a^\epsilon/t\}} \parallel \\
 H\{t\} \\
 \Psi \parallel \\
 A
 \end{array}
 ;$$

(3)

$$\begin{array}{c}
 \Pi'' \parallel \\
 K \left\{ \frac{a \vee a}{a^\epsilon} \right\} \\
 \Phi \parallel \\
 H \left\{ \frac{a^\epsilon}{t} \right\} \\
 \Psi \parallel \\
 A
 \end{array}
 \text{ becomes }
 \begin{array}{c}
 \Pi'' \parallel \\
 K \left[\frac{a}{\frac{t}{t} \vee \frac{a}{t}} \right] \\
 \Phi_{\{a^\epsilon/t\}} \parallel \\
 H\{t\} \\
 \Psi \parallel \\
 A
 \end{array}
 ;$$

(4)

$$\begin{array}{c}
 \Pi'' \parallel \\
 K \left\{ \frac{a}{a^\epsilon \wedge a} \right\} \\
 \Phi \parallel \\
 H \left\{ \frac{a^\epsilon}{t} \right\} \\
 \Psi \parallel \\
 A
 \end{array}
 \text{ becomes }
 \begin{array}{c}
 \Pi'' \parallel \\
 K\{a\} \\
 \Phi_{\{a^\epsilon/t\}} \parallel \\
 H\{t\} \\
 \Psi \parallel \\
 A
 \end{array}
 .$$

The process terminates in linear time on the size of Π' because each transformation eliminates some atom occurrences. The final proof is in aSKS. \square

The transformations described in the proof of Theorem 27 are the minimal ones necessary to produce a proof in aSKS. However, it is possible to further reduce the proof so obtained. The transformations in the proof of Theorem 27, together with the one mentioned in Step (1) in the proof of Theorem 11, all belong to the class of weakening and

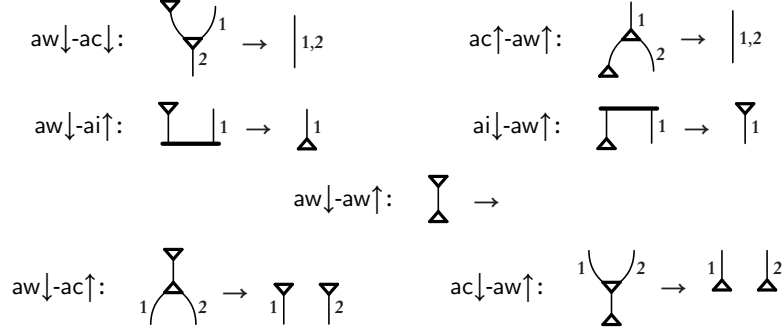
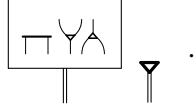


FIGURE 6. Weakening and coweakening atomic-flow reductions.

coweakening reductions studied in [GG08]. In the rest of this section, we quickly outline a possible, further transformation of the analytic form produced by those reductions, and refer the reader to [GG08] for a more thorough explanation.

It is advantageous to describe the weakening and coweakening transformations directly as atomic-flow reduction rules. These are special graph rewriting rules for atomic flows, that are known to correspond to sound derivation transformations, in the following sense. If Φ is a derivation with flow ϕ , and ϕ can be transformed into ψ by one of the atomic-flow reduction rules, then there exists a derivation Ψ whose flow is ψ and such that it has the same premiss and conclusion as Φ . Moreover, Ψ can be obtained from Φ by instantiating some atoms and changing some rule instances, in linear time.

The weakening and coweakening atomic-flow reduction rules are shown in Figure 6. The reduction rule labelled $\text{aw}\downarrow\text{-ai}\uparrow$ is employed in Step (1) in the proof of Theorem 11. The reduction rules labelled $\text{ac}\uparrow\text{-aw}\uparrow$, $\text{ai}\downarrow\text{-aw}\uparrow$, $\text{aw}\downarrow\text{-aw}\uparrow$ and $\text{ac}\downarrow\text{-aw}\uparrow$ are employed in the proof of Theorem 27, respectively as Case (4), (1), (2) and (3). If we apply the full set of weakening and coweakening reductions until possible, starting from a proof in cut-free form, we obtain a proof of the same formula and whose flow has shape



Note that the graph rewriting system consisting of the reductions in Figure 6 is confluent.

8. FINAL COMMENTS AND FUTURE WORK

System aSKS is not a minimal complete system for propositional logic, because the atomic cocontraction rule $\text{ac}\uparrow$ is admissible (via $\text{ac}\downarrow$, s , $\text{ai}\uparrow$ and cut elimination). Removing $\text{ac}\uparrow$ from aSKS yields system KS. A natural question is whether quasipolynomial-time normalisation holds for KS as well. We would guess that cocontraction plays an essential role in keeping the complexity low. For example, one can note in Figure 5 how cocontraction limits the size of the n pieces of derivation below each θ_k . If we had to expand those cocontraction instances into a tree we would have an exponential blow-up. On the other hand, an encouraging result in the opposite direction is contained in [Das14], where the author obtains $n^{O(\log \log n)}$ -size proofs of the weak pigeonhole principle, using deep-inference techniques to improve the previous bound for monotone proofs.

There is reason to believe that polynomial normalisation is achievable, because it is possible to compute threshold functions with polynomial formulae. However, the hardest problem seems to be obtaining polynomial Γ -like (cut-free) derivations with the property of Theorem 21. We tend to think that polynomiality ought to be possible, and deep

inference might be a helpful language for investigating and achieving it, because of its flexibility in constructing derivations.

The normalisation procedure presented here is peculiar because it achieves its result by using an external scheme, constituted by the threshold formulae and the Γ derivations, which does not depend on the derivation to be normalised. Threshold formulae realise a clever compositional mechanism built on top of cocontractions. It would be interesting to interpret this or a similar mechanism computationally. We do not necessarily expect an interpretation strictly following the Curry-Howard scheme, and there is little evidence that the threshold construction studied here can be applied in the intuitionistic case. On the other hand an essential ingredient of our construction, namely sharing by cocontractions, already allowed us to reach beyond the limitations of Gentzen-style proof theory: in recent work fully lazy sharing has been achieved in a λ -calculus within intuitionistic deep inference [GHP13a, GHP13b]. For the authors of this paper and for some of their colleagues this is an active research area.

It is possible to extend the mechanism investigated here to the more general notion of normalisation that we called *streamlining* in [GG08]; this has been done in [Gun09]. Streamlining is a top-down symmetric notion, that does full justice to the additional symmetry of deep inference, compared to Gentzen formalisms. Streamlined derivations entail analytic proofs as a special case.

The results of this paper are, as mentioned previously, closely related to results about the monotone sequent calculus MLK, through the translation between SKS and LK given in [Brü06]. Atserias, Galesi and Pudlák show in [AGP02] that MLK can quasipolynomially simulate LK over monotone sequents, and as shown by Jeřábek in [Jeř09], this implies Theorem 24.

Furthermore, in [Jeř12], Jeřábek considers a conservative extension of MLK, called MCLK, and shows how MCLK can quasipolynomially simulate LK over arbitrary formulae. MCLK is defined by restricting LK to only allow cuts on monotone formulae. Like MCLK, streamlined derivations are also a conservative extension of MLK and the two notions are very similar. Exploring this will be the subject of future work.

Finally, we are interested in the normalisation theory of modal logics in deep inference, and so we are naturally led to consider the methods presented in this paper to that purpose as well.

REFERENCES

- [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.
- [BG09] Paola Bruscoli and Alessio Guglielmi. On the proof complexity of deep inference. *ACM Transactions on Computational Logic*, 10(2):14:1–34, 2009.
- [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.
- [Brü04] Kai Brünnler. *Deep Inference and Symmetry in Classical Proofs*. Logos Verlag, Berlin, 2004.
- [Brü06] Kai Brünnler. Deep inference and its normal form of derivations. In Arnold Beckmann, Ulrich Berger, Benedikt Löwe, and John V. Tucker, editors, *Logical Approaches to Computational Barriers—2nd Conference on Computability in Europe*, volume 3988 of *Lecture Notes in Computer Science*, pages 65–74. Springer-Verlag, 2006.
- [BT01] Kai Brünnler and Alwen Fernanto Tiu. A local system for classical logic. In R. Nieuwenhuis and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning (LPAR)*, volume 2250 of *Lecture Notes in Computer Science*, pages 347–361. Springer-Verlag, 2001.
- [Bus91] Samuel R. Buss. The undecidability of k-provability. *Annals of Pure and Applied Logic*, 53(1):75–102, 1991.
- [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.

- [Das14] Anupam Das. On the pigeonhole and related principles in deep inference and monotone systems. In Thomas Henzinger and Dale Miller, editors, *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)*, pages 36:1–10. ACM, 2014.
- [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.
- [GGP10] Alessio Guglielmi, Tom Gundersen, and Michel Parigot. A proof calculus which reduces syntactic bureaucracy. In Christopher Lynch, editor, *21st International Conference on Rewriting Techniques and Applications (RTA)*, volume 6 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 135–150. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2010.
- [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 (LICS)*, pages 284–293. IEEE, 2010.
- [GHP13a] Tom Gundersen, Willem Heijltjes, and Michel Parigot. Atomic lambda calculus: A typed lambda-calculus with explicit sharing. In Orna Kupferman, editor, *28th Annual IEEE Symposium on Logic in Computer Science (LICS)*, pages 311–320. IEEE, 2013.
- [GHP13b] Tom Gundersen, Willem Heijltjes, and Michel Parigot. A proof of strong normalisation of the typed atomic lambda-calculus. In Ken McMillan, Aart Middeldorp, and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning (LPAR-19)*, volume 8312 of *Lecture Notes in Computer Science*, pages 340–354. Springer-Verlag, 2013.
- [Gug] Alessio Guglielmi. Deep inference. Web site at <http://alessio.guglielmi.name/res/cos>.
- [Gug07] Alessio Guglielmi. A system of interaction and structure. *ACM Transactions on Computational Logic*, 8(1):1:1–64, 2007.
- [Gug09] Alessio Guglielmi. Th.pl, 2009. Prolog program, available at <http://cs.bath.ac.uk/ag/p/th.pl>.
- [Gun09] Tom Gundersen. *A General View of Normalisation Through Atomic Flows*. PhD thesis, University of Bath, 2009.
- [Jeř09] Emil Jeřábek. Proof complexity of the cut-free calculus of structures. *Journal of Logic and Computation*, 19(2):323–339, 2009.
- [Jeř12] Emil Jeřábek. Proofs with monotone cuts. *Mathematical Logic Quarterly*, 58(3):177–187, 2012.
- [Sta78] Richard Statman. Bounds for proof-search and speed-up in the predicate calculus. *Annals of Mathematical Logic*, 15:225–287, 1978.
- [Weg87] Ingo Wegener. *The Complexity of Boolean Functions*. John Wiley & Sons Ltd and B. G. Teubner, Stuttgart, 1987.

BRUSCOLI, GUGLIELMI: UNIVERSITY OF BATH (UK).

GUNDERSEN: RED HAT, INC.

PARIGOT: LABORATOIRE PPS, UMR 7126, CNRS & UNIVERSITÉ PARIS 7 (FRANCE).