Combinatorial Flows and Proof Compression

Lutz Straßburger

Combinatorial Flows and Proof Compression

Lutz Straßburger

RESEARCH REPORT
N° 9048
March 2017
Project-Team Parsifal

ISSN 0249-6399   ISRN INRIA/RR--9048--FR+ENG

Combinatorial Flows and Proof Compression

Lutz Straßburger

Project-Team Parsifal

Research Report n° 9048 — March 2017 — 16 pages

Abstract: This paper introduces the notion of combinatorial flows as a generalization of combinatorial proofs that also includes cut and substitution as methods of proof compression. We show a normalization procedure for combinatorial flows, and how syntactic proofs in sequent calculus, deep inference, and Frege systems are translated into combinatorial flows and vice versa.

Key-words: Combinatorial flows, deep inference, proof compression, cut elimination, substitution elimination

RESEARCH CENTRE
SACLAY – ÎLE-DE-FRANCE

1 rue Honoré d'Estienne d'Orves
Bâtiment Alan Turing
Campus de l'École Polytechnique
91120 Palaiseau

Fleuves combinatoires et compression des preuves

Résumé : Cet article introduit la notion de fleuves combinatoires comme généralisation des preuves combinatoires qui comprend également la coupure et la substitution comme méthodes de compression des preuves. Nous montrons un procédure de normalisation des fleuves combinatoires, et la traduction entre les preuves syntactiques du calcul de séquents, de l'inférence profonde, des systèmes de Frege, et les fleuves combinatoires.

Mots-clés : fleuves combinatoires, inference profonde, compression des preuves, élimination des coupures

1. Introduction

Proof theory is a central area of theoretical computer science, as it can provide the foundations not only for logic programming and functional programming, but also for the formal verification of software. Yet, despite the crucial role played by formal proofs, we have no proper notion of proof identity telling us when two proofs are "the same". This is very different from other areas of mathematics, like group theory, where two groups are "the same" if they are isomorphic, or topology, where two spaces are "the same" if they are homeomorphic.

The problem is that proofs are usually presented by syntactic means, and depending on the chosen syntactic formalism, "the same" proof can look very different. In fact, one can say that at the current state of art, *proof theory is not a theory of proofs but a theory of proof systems*. This means that the first step must be to find ways to describe proofs independent from the proof systems. In other words, we need a "syntax-free" presentation of proofs.

Syntax-free presentations of proofs

The earliest attempts for such "syntax-free" proof presentations were Andrews' *matings* [And76] and Bibel's *matrix proofs* [Bib81] for propositional logic. However, checking correctness of a mating or matrix proof is exponential, and thus not more efficient than starting a proof search from scratch. Furthermore, matings and matrix proofs are not able to address proof normalization procedures like cut elimination.

The first notion of syntax-free proof presentation that was able to address these two issues are Girard's *proof-nets* for linear logic [Gir87], which are graphs that abstract away from the syntax of the sequent calculus, such that it is decidable in polynomial time whether a given such graph is indeed a correct proof, and such that the normalization of proofs via cut elimination is simpler in proof-nets than in the sequent calculus.

Clearly, it became a research question whether such a notion of proof-net is also possible for classical logic. An immediate idea is to use exactly the same notion of proof-net as for linear logic [Lau99], [Lau03], [Rob03]. However, these proof-nets depend on Gentzen's sequent calculus LK [Gen34]. They are neither able to capture proofs written in other sequent calculi, like G3c [TS00], nor other formalisms, like analytic tableaux or resolution.

This problem was addressed by *B-nets* [LS05], which exhibit a confluent cut elimination procedure and can capture proofs in most standard proof formalisms. However, their correctness criterion is exponential and the cut elimination cannot be lifted to the sequent calculus.

This issue has been addressed by *atomic flows* [GG08], [GGS10] that are more fine-grained than Boolean nets and that have a number of different cut elimination procedures that can all be lifted to a deep inference proof system. However, atomic flows do not have a correctness criterion. In fact, the work by Das [Das13] shows that there cannot be a polynomial correctness criterion for atomic flows, if integer factoring is hard for *P/poly*. The same problem have the so-called *C-nets* [Str11] which are similar to atomic flows, but additionally form a closed category.

Only the *combinatorial proofs* by Hughes [Hug06a] have a polynomial correctness criterion and are independent from any syntactic formalism. But they have no notion of proof composition—they are inherently cut-free. The only way to speak about cut in combinatorial flows is to add additional formulas $A \wedge \bar{A}$ to the conclusion [Hug06b].

Methods of proof compression

The *cut* can be seen as a method of proof compression, in the sense that a proof with cuts is in general much smaller than the proof that is obtained by elimination the cuts [Boo84]. In the area of structural proof theory this is a well studied and well understood phenomenon. However, there are other methods of proof compression, namely *extension* and *substitution* [CR79], that are mainly studied in the area of proof complexity. In fact, it is one of the major open problems of proof complexity whether Frege systems without extension can p-simulate Frege-systems with extension. But from the viewpoint of structural proof theory, the notions of extension and substitution have not yet been much investigated.

It has long been known that in the presence of cut, extension and substitution have the same strength with respect to p-simulation (shown in [CR79] and [KP89]). But only recently is has been shown that also in the absence of cut, systems with extension and systems with substitution can p-simulate each other [Str12], [NS15]. For this, a deep inference proof systems has been used that can speak about cut elimination and extension elimination at the same time so that the two proof compression mechanisms can be studied together. For Frege-systems, which are the ordinary vehicle for studying extension and substitution, there is no "cut-free" version.

However, so far, there is no "syntax-free" proof presentation that can deal with proofs using extension or substitution.

Contributions and outline of this paper

The main contribution of this paper is a notion of "syntax-free" proof presentation that comes with a polynomial correctness criterion, that is independent of the syntax of proof formalisms (like sequent calculi, tableaux systems, resolution, Frege systems, or deep inference systems) and that can handle cut and substitution, and their elimination. The main idea is to combine the advantages of combinatorial proofs and of atomic flows, and add a notion of substitution.

The paper is subdivided into three parts:

1) First, in Sections 2 and 3, we recall the preliminaries on combinatorial proofs and then present an up-down symmetric variant, that we will call *simple combinatorial flows*. Then we define formal operations on combinatorial flows: (i) *horizontal composition* via the binary connectives $\wedge$ and $\vee$, (ii) *vertical composition* via a *cut*, and (iii) *substitution* of one proof into another. This last operation is more general than just substituting formulas into formulas. The first immediate observation is that combinatorial flows form a proof system in the sense of Cook and Reckhow [CR79].

2) Then, in Sections 4–7, we will study the normalization of combinatorial flows. A combinatorial flow is *normal* iff it is a simple combinatorial flow, i.e., does not contain any of the operations mentioned before. We will see in Section 4 that the elimination of horizontal composition does not yield an increase in the size of the flow. But the elimination of *substitution* (in Section 5) and of *cut* (in Section 6) will each cause an exponential blow-up of the size of the flow.
3) Finally, in Sections 8 and 9, we will see how we can translate between combinatorial flows and syntactic proofs in ordinary deductive systems. We will cover sequent calculus, deep inference, and Frege systems. The expected results are:
   - Simple combinatorial flows are p-equivalent to cut-free sequent calculus.
   - Simple combinatorial flows correspond to deep inference proofs in a certain decomposition normal form.
   - Combinatorial flows with cut but without substitution are p-equivalent to sequent calculus with cut, to the symmetric deep inference system SKS [BT01], and to ordinary Frege systems.
   - Finally, combinatorial flows with cut and substitution are p-equivalent to SKS with extension or substitution and to Frege systems with extension or substitution.

2. Preliminaries on combinatorial proofs

Combinatorial proofs have been introduced by Hughes in [Hug06a] as a way to present proofs of classical logic independent from a syntactic proof system. To make our paper self-contained, we recall here the basic definitions.

We consider formulas (denoted by capital Latin letters $A, B, C, \ldots$) in negation normal form (NNF), generated from a countable set $\mathcal{V} = \{a, b, c, \ldots\}$ of (propositional) variables by the following grammar:

$$A, B ::= a \mid \bar{a} \mid A \wedge B \mid A \vee B \qquad (1)$$

where $\bar{a}$ is the negation of a. The negation can then be defined for all formulas using the De Morgan laws:

$$\bar{\bar{A}} = A \qquad \overline{A \wedge B} = \bar{A} \vee \bar{B} \qquad (2)$$

An *atom* is a variable or its negation. We use $\mathcal{A}$ to denote the set of all atoms. Sometimes we use $A \Rightarrow B$ as abbreviation for $\bar{A} \vee B$, and $A \Leftrightarrow B$ as abbreviation for $(A \Rightarrow B) \wedge (B \Rightarrow A)$.

A *sequent* Γ is a multiset of formulas, written as a list separated by comma:

$$\Gamma = A_1, A_2, \ldots, A_n \qquad (3)$$

We write $\bar{\Gamma}$ to denote the sequent $\bar{A}_1, \bar{A}_2, \ldots, \bar{A}_n$. We define the *size* of a sequent Γ, denoted by $|\Gamma|$, to be the number of atom occurrences in it. We write $\bigwedge\Gamma$ (resp. $\bigvee\Gamma$) for the conjunction (res. disjunction) of the formulas in Γ.

Remark 2.1. For simplicity we do not include the constants $\top$ and $\bot$ (for *truth* and *falsum*, respectively) into the language. We can always recover them by letting $\top = a_0 \vee \bar{a}_0$ and $\bot = a_0 \wedge \bar{a}_0$ for some fresh variable a_0. Note that in this respect, classical logic is different from linear logic, where the removal of the constants does indeed change the logic.

Definition 2.2. A *(simple) graph* $\mathfrak{G} = \langle V_\mathfrak{G}, E_\mathfrak{G} \rangle$ consists of a set of *vertices* $V_\mathfrak{G}$ and a set of *edges* $E_\mathfrak{G}$ which are two-element subsets of $V_\mathfrak{G}$. If $E_\mathfrak{G}$ is not a set but a multiset, we call $\mathfrak{G}$ a *multigraph*. We omit the index $\mathfrak{G}$ when it is clear from context. For $v, w \in V$ we write vw for $\{v, w\}$. The *size* of a graph $\mathfrak{G}$, denoted by $|\mathfrak{G}|$ is $|V_\mathfrak{G}| + |E_\mathfrak{G}|$. A *graph homomorphism* $f \colon \mathfrak{G} \to \mathfrak{G}'$ is a function from $V_\mathfrak{G}$ to $V_{\mathfrak{G}'}$ such that $vw \in E_\mathfrak{G}$ implies $f(v)f(w) \in E_{\mathfrak{G}'}$. A simple graph $\mathfrak{G}$ is called a *cograph* if it does not contain four distinct vertices u, v, w, z with $uv, vw, wz \in E$ and $vz, zu, uw \notin E$. For a set L, a graph $\mathfrak{G}$ is *L-labeled* if every vertex of $\mathfrak{G}$ is associated with an element L, called its *label*. For two graphs $\mathfrak{G} = \langle V, E \rangle$ and $\mathfrak{G}' = \langle V', E' \rangle$, we define the operations *union* $\mathfrak{G} \vee \mathfrak{G}' = \langle V \cup V', E \cup E' \rangle$ and *join* $\mathfrak{G} \wedge \mathfrak{G}' = \langle V \cup V', E \cup E' \cup \{vv' \mid v \in V, v' \in V'\} \rangle$. If $\mathfrak{G}$ and $\mathfrak{G}'$ are L-label-led graphs, then so are $\mathfrak{G} \vee \mathfrak{G}'$ and $\mathfrak{G} \wedge \mathfrak{G}'$ where every vertex keeps its original label. For a simple graph $\mathfrak{G} = \langle V, E \rangle$, also define its *negation* $\bar{\mathfrak{G}} = \langle V, \{vw \mid v \neq w, vw \notin E\} \rangle$. If $\mathfrak{G}$ is an $\mathcal{A}$-labeled graph (where $\mathcal{A}$ is the set of atoms) then all labels are negated in $\bar{\mathfrak{G}}$. For two homomorphisms $f_1 \colon \mathfrak{G}_1 \to \mathfrak{G}'_1$ and $f_2 \colon \mathfrak{G}_2 \to \mathfrak{G}'_2$ such that $V_{\mathfrak{G}_1} \cap V_{\mathfrak{G}_2} = \emptyset$, we define $f_1 \vee f_2 \colon \mathfrak{G}_1 \vee \mathfrak{G}_2 \to \mathfrak{G}'_1 \vee \mathfrak{G}'_2$ to be the *union* of the two homomorphisms f_1 and f_2, and $f_1 \wedge f_2 \colon \mathfrak{G}_1 \wedge \mathfrak{G}_2 \to \mathfrak{G}'_1 \wedge \mathfrak{G}'_2$ to be their *join*.

Construction 2.3. If we associate to each atom a a single vertex labeled with a then every formula A uniquely determines a graph $\mathfrak{G}(A)$ that is constructed via the operations $\wedge$ and $\vee$. For a sequent $\Gamma = A_1, A_2, \ldots, A_n$, we define

$$\mathfrak{G}(\Gamma) = \mathfrak{G}(\bigvee\Gamma) = \mathfrak{G}(A_1) \vee \mathfrak{G}(A_2) \vee \cdots \vee \mathfrak{G}(A_n) \quad .$$

Note that this construction entails that $\overline{\mathfrak{G}(A)} = \mathfrak{G}(\bar{A})$.

Lemma 2.4. *For two formulas A and B, we have $\mathfrak{G}(A) = \mathfrak{G}(B)$ iff A and B are equivalent modulo associativity and commutativity of $\wedge$ and $\vee$:*

$$\begin{array}{ll} A \wedge (B \wedge C) = (A \wedge B) \wedge C & A \wedge B = B \wedge A \\ A \vee (B \vee C) = (A \vee B) \vee C & A \vee B = B \vee A \end{array} \qquad (4)$$

Proof. Immediately from Construction 2.3. □

Example 2.5. Let $A = (a \wedge (b \vee \bar{c})) \vee (c \wedge \bar{d})$ then $\bar{A} = (\bar{a} \vee (\bar{b} \wedge c)) \wedge (\bar{c} \vee d)$. Below are the two graphs $\mathfrak{G}(A)$ and $\mathfrak{G}(\bar{A}) = \overline{\mathfrak{G}(A)}$:

(5)

The following is well-known. It can already be found in [Duf65], see also [Möh89], [Ret93].

Proposition 2.6. *A graph $\mathfrak{G}$ is a cograph iff it is constructed from a formula via Construction 2.3.*

An important consequence of this and Lemma 2.4 is that for each cograph $\mathfrak{G}$ there is a unique (up to associativity and commutativity) formula tree determining $\mathfrak{G}$. We denote this formula tree by $F(\mathfrak{G})$.

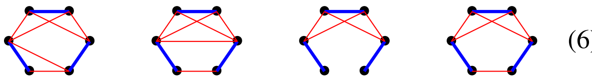
Definition 2.7. Let $\mathfrak{G} = \langle V, E \rangle$ be a cograph, let $V' \subseteq V$, and let E' be the restriction of E to V'. We say that

$\mathfrak{G}' = \langle V', E' \rangle$ is a *subcograph* of $\mathfrak{G}$ iff for all $v \in V'$ and $w_1, w_2 \in V \backslash V'$ we have $vw_1 \in E$ iff $vw_2 \in E$. In this case we also say that V' *induces a subcograph*.

It follows immediately from the definition that any subcograph is indeed a cograph. Furthermore, $\mathfrak{G}'$ is a subcograph of $\mathfrak{G}$ iff $F(\mathfrak{G}')$ is a subformula of $F(\mathfrak{G})$.

Definition 2.8. Let $\mathfrak{G} = \langle V, E \rangle$ be a multigraph. A set $B \subseteq E$ of edges is called a *matching* if no two edges in B are adjacent. A matching B is *perfect* if every vertex $v \in V_{\mathfrak{G}}$ is incident to an edge in B. An *R&B-graph* $\mathfrak{G} = \langle V, R, B \rangle$ is a triple such that $\langle V, R \uplus B \rangle$ is a multigraph such that B is a perfect matching and $\langle V, R \rangle$ is a simple graph (i.e., R is not allowed to have multiple edges). We will use the notation $\mathfrak{G}^{\downarrow}$ for the simple graph $\langle V, R \rangle$. An *R&B-cograph* is an R&B-graph $\mathfrak{G} = \langle V, R, B \rangle$ where $\mathfrak{G}^{\downarrow} = \langle V, R \rangle$ is a cograph.

Following [Ret03] we will draw B-edges in blue/bold, and R-edges in red/regular. Below are four examples:



(6)

Also the following two definitions are taken from [Ret03].

Definition 2.9. A path (resp. cycle) in a graph is said to be *elementary* if it does not contain two equal vertices (resp. but the first and last one). A path $\mathcal{P}$ in a graph with a matching B is *alternating* if the edges of $\mathcal{P}$ are alternately in B and not in B. Let $\mathfrak{G} = \langle V, R, B \rangle$ be an R&B-graph. An *æ-path* in $\mathfrak{G}$ is an elementary alternating path in $\langle V, R \uplus B \rangle$. An *æ-cycle* in $\mathfrak{G}$ is an elementary alternating cycle of even length in $\langle V, R \uplus B \rangle$, so that when turning around the cycle, the edges are still alternately in B and not in B. A *chord* of a path (resp. cycle) is an edge that is not part of the path (resp. cycle) but connects two vertices of the path (resp. cycle). An æ-path (resp. æ-cycle) is called *chordless* iff it does not have any chords.

Note that chords for æ-paths, resp. æ-cycles, are always R-edges because B is a perfect matching. We are now ready to present a central concept for R&B-cographs:

Definition 2.10. An R&B-cograph $\mathfrak{G} = \langle V, R, B \rangle$ is *critically chorded* if $\langle V, R \uplus B \rangle$ does not contain any chordless æ-cycle, and any two vertices in V are connected by a chordless æ-path.

In the examples in (6), the first one is not an R&B-cograph, the other three are. The second one has a chordless æ-cycle, and the third one has no chordless æ-path between the lowermost vertices. Only the last one is a critically chorded R&B-cograph.

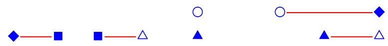
Definition 2.11. Let $\mathfrak{C} = \langle V, R, B \rangle$ be an R&B-graph and $f \colon \mathfrak{C}^{\downarrow} \to \mathfrak{G}$ be a graph-homomorphism and let $\mathfrak{G}$ be $\mathcal{A}$-labeled (where $\mathcal{A}$ is the set of atoms). We say f is *axiom-preserving* iff $xy \in B$ implies that the labels of $f(w)$ and $f(v)$ are dual to each other.

Definition 2.12. A graph homomorphism f is a *skew fibration*, denoted as $f \colon \mathfrak{G} \rightarrowtail \mathfrak{G}'$, if for every $v \in V_{\mathfrak{G}}$ and $w' \in V_{\mathfrak{G}'}$ with $f(v)w' \in E'_{\mathfrak{G}}$ there is a $w \in V_{\mathfrak{G}}$ with $vw \in E_{\mathfrak{G}}$ and $f(w)w' \notin E'_{\mathfrak{G}}$.

We are now ready to give the definition of a combinatorial proof together with the main result of [Hug06a]

Definition 2.13. A *combinatorial proof* of a sequent Γ consists of a non-empty critically chorded R&B-cograph $\mathfrak{C}$ and an axiom-preserving skew-fibration $f \colon \mathfrak{C}^{\downarrow} \rightarrowtail \mathfrak{G}(\Gamma)$.

In [Hug06a] and [Hug06b], combinatorial proofs are depicted by writing the sequent Γ and drawing the graph $\mathfrak{C}$ above Γ such that for each atom occurrence in Γ, the vertices of $\mathfrak{C}$ that map to it are drawn directly above it. Then the R-edges are shown, but the B-edges are indicated by using different shapes for drawing the vertices. Here is an example



$$\bar{a} \wedge c,\ (\bar{c} \wedge \bar{a}) \vee \bar{b} \vee c,\ (b \vee b) \wedge a \tag{7}$$

Remark 2.14. Our presentation of the condition on the cograph in a combinatorial proof deters from Hughes' [Hug06a] and follows Retoré's [Ret03] instead. The reason is that Retoré makes the relation to proof nets of linear logic [DR89] explicit. Also note, that the condition on the cograph $\mathfrak{C}^{\downarrow}$ given by Hughes [Hug06a], [Hug06b] is weaker than ours. It is equivalent to our condition of $\mathfrak{C}$ not containing any chordless æ-cycle. In terms of linear logic, this is equivalent to the correctness condition for MLL proof nets with the mix-rule [Ret03]. In our presentation here we also add the connectedness via chordless æ-paths in order to reject mix. *A priori*, for classical logic it is irrelevant whether mix (which says that $A \wedge B$ implies $A \vee B$) is allowed or not since it is derivable using weakening. However, we can obtain stronger results (in particular the Decomposition Theorem 8.2 in Section 8) if we reject mix.

The first result of [Hug06a] is that combinatorial proofs are sound and complete with respect to classical logic.

Theorem 2.15 ([Hug06a])**.** *A formula is a theorem of classical propositional logic iff it has a combinatorial proof.*

The proof given in [Hug06a] is based on semantics, and it works equally well with our stronger criterion. But it is also possible to give a syntactic proof: in [Hug06b], Hughes shows how a sequent calculus proof can be translated into a combinatorial proof, which immediately entails completeness. Then, as mentioned above, a critically chorded R&B-cograph corresponds to a proof in multiplicative linear logic [Ret03], and a skew-fibration corresponds to precisely the maps that can be constructed from contraction and weakening [Hug06b], [Str07]. This entails soundness.

The second important result of [Hug06a] is that combinatorial proofs form a proof system in the sense of Cook and Reckhow [CR79].

Theorem 2.16 ([Hug06a])**.** *Combinatorial proofs form a proof system.*

Proof (Sketch). We have to show that, given some formula A, some R&B-graph $\mathfrak{C}$, and some mapping $f \colon \mathfrak{C}^{\downarrow} \to \mathfrak{G}(A)$, we can decide in polynomial time that

- $\mathfrak{C}^{\downarrow}$ is a cograph,

- $\mathfrak{C}$ is critically chorded,
- f is axiom-preserving, and
- f is a skew-fibration.

The first can be done in $O(|V_{\mathfrak{C}}| + |R_{\mathfrak{C}}|)$ as shown in [CPS85]. The second is equivalent to correctness of MLL-proof nets (shown in [Ret03]), which can be checked in linear time (shown in [Gue99]). The last two conditions on f are trivially polynomial. □

3. Combinatorial flows

Definition 3.1. Given two sequents Γ and Δ, a *simple (combinatorial) flow* ϕ from Γ to Δ, denoted by $\phi\colon \Gamma \vdash \Delta$, is a combinatorial proof for the sequent $\bar{\Gamma}, \Delta$. We write $\phi\colon \circ \vdash \Delta$ (resp. $\phi\colon \Gamma \vdash \circ$) if Γ (resp. Δ) is empty.[1] Let ϕ be given by the R&B-cograph $\mathfrak{C}$ and skew fibration $f\colon \mathfrak{C}^{\downarrow} \rightarrowtail \mathfrak{G}(\bar{\Gamma}, \Delta)$. Then the *size* of ϕ, denoted by $|\phi|$, is defined to be $|\mathfrak{C}^{\downarrow}| + |\Gamma| + |\Delta|$.

Theorem 3.2. *Simple combinatorial flows form a proof system.*

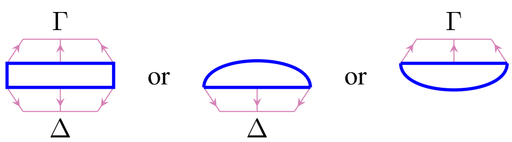
Proof. This follows immediately from Theorem 2.16 and Definition 3.1. □

Lemma 3.3. *Let $\mathfrak{C}$, $\mathfrak{G}_1$, and $\mathfrak{G}_2$ be cographs and let $f\colon \mathfrak{C} \rightarrowtail \mathfrak{G}_1 \vee \mathfrak{G}_2$ be a skew fibration. Then there are cographs $\mathfrak{C}_1$ and $\mathfrak{C}_2$ and graph homomorphisms $f_1\colon \mathfrak{C}_1 \rightarrowtail \mathfrak{G}_1$ and $f_2\colon \mathfrak{C}_2 \rightarrowtail \mathfrak{G}_2$ such that $\mathfrak{C} = \mathfrak{C}_1 \vee \mathfrak{C}_2$ and $f = f_1 \vee f_2$.*

Proof. This follows immediately from f being a homomorphism. We can let $V_{\mathfrak{C}_1}$ and $V_{\mathfrak{C}_2}$ be the inverse images of $V_{\mathfrak{G}_1}$ and $V_{\mathfrak{G}_2}$, respectively, under f, and let $\mathfrak{C}_1$ and $\mathfrak{C}_2$ be the induced subgraphs. □

Notation 3.4. This lemma allows us to depict basic combinatorial flows in the following way. Let $\phi\colon \Gamma \vdash \Delta$ be given, let $f\colon \mathfrak{C}^{\downarrow} \rightarrowtail \overline{\mathfrak{G}(\Gamma)} \vee \mathfrak{G}(\Delta)$ be the defining skew fibration, and let $\mathfrak{C}_\Gamma$ and $\mathfrak{C}_\Delta$ be the cographs determined by Lemma 3.3 (i.e., $\mathfrak{C}^{\downarrow} = \mathfrak{C}_\Gamma \vee \mathfrak{C}_\Delta$). If we write $F(\overline{\mathfrak{C}_\Gamma})$ and $F(\mathfrak{C}_\Delta)$ for the formula trees corresponding to the cographs $\overline{\mathfrak{C}_\Gamma}$ and $\mathfrak{C}_\Delta$, respectively, then we can write ϕ by writing Γ, $F(\overline{\mathfrak{C}_\Gamma})$, $F(\mathfrak{C}_\Delta)$, and Δ above each other, draw the B-edges and indicate the mapping f by thin (thistle) arrows. Figure 1 shows some examples. For better readability, we allow in $F(\overline{\mathfrak{C}_\Gamma})$ outermost $\wedge$ to be replaced by comma, and in $F(\mathfrak{C}_\Delta)$ outermost $\vee$ to be replaced by comma. Note that the first three flows in Figure 1 are just "flipped variants" of each other, i.e., are defined by the same R&B-cograph and skew fibration; the first one being the same as the combinatorial proof in (7).

Schematically we can depict simple combinatorial flows as follows:



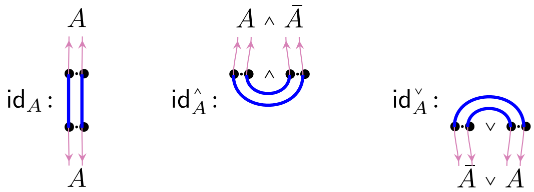
where the middle and the right picture are used to indicate that Γ or Δ, respectively, are empty.

1. Note that it cannot happen that both Γ and Δ are empty.

Lemma 3.5. *Let Γ, Δ, Σ be sequents. There is a one-to-one correspondence between the simple combinatorial flows $\Gamma \vdash \Sigma, \Delta$ and $\bar{\Sigma}, \Gamma \vdash \Delta$. In particular, for any three formulas A, B, C, there is a one-to-one correspondence between the simple combinatorial flows $A \vdash B \vee C$ and $\bar{B} \wedge A \vdash C$.*

Proof. This follows immediately from Definition 3.1. □

Observation 3.6. For every formula A, we have a simple combinatorial flow $\mathsf{id}_A\colon A \vdash A$, that we call the *identity flow* and that is defined by the identity skew fibration $\overline{\mathfrak{G}(A)} \vee \mathfrak{G}(A) \rightarrowtail \mathfrak{G}(\bar{A}, A)$ where the matching is defined such that it pairs each vertex in $V_{\mathfrak{G}(A)}$ to itself in the copy $V_{\overline{\mathfrak{G}(A)}}$. When applying Lemma 3.5 to id_A we get two simple combinatorial flows $\mathsf{id}^{\wedge}_A\colon A \wedge \bar{A} \vdash \circ$ and $\mathsf{id}^{\vee}_A\colon \circ \vdash \bar{A} \vee A$, as depicted below:



(8)

Definition 3.7. A *substitution* is a mapping σ from propositional variables to formulas such that $\sigma(a) \neq a$ for only finitely many a.

We write $A\sigma$ for the formula obtained from applying the substitution σ to the formula A. If $\sigma = \{a_1 \mapsto B_1, \ldots, a_n \mapsto B_n\}$ we also write $A[a_1/B_1, \ldots, a_n/B_n]$ for $A\sigma$. This normally means that not only is each occurrence of a_i in A is replaced by B_i in A, but also each occurrence of $\bar{a}_i$ is replaced by $\bar{B}_i$. We also need a notation for substitutions in which an variable a and its dual $\bar{a}$ are not replaced by dual formulas. In this case we write $A[a_1/B_1, \bar{a}_1/C_1, \ldots, a_n/B_n, \bar{a}_n/C_n]$ for the formula that is obtained from A by simultaneously replacing every a_i by B_i and every $\bar{a}_i$ by C_i for each $i \in \{1, \ldots, n\}$.

Definition 3.8. The set of *combinatorial flows* is defined inductively as follows:

- A simple combinatorial flow $\phi\colon A \vdash B$ is a combinatorial flow.
- If $\phi\colon A \vdash B$ and $\psi\colon C \vdash D$ are combinatorial flows then so are $\phi \wedge \psi\colon A \wedge B \vdash C \wedge D$ and $\phi \vee \psi\colon A \vee C \vdash B \vee D$. This operation is called *horizontal composition*.
- If $\phi\colon \Gamma \vdash A$ and $\psi\colon A \vdash \Delta$ are combinatorial flows then $\phi \diamond \psi\colon \Gamma \vdash \Delta$ is a combinatorial flow. This operation is called *vertical composition*, *concatenation*, or *cut*.
- If $\phi\colon \Gamma \vdash \Delta$ and $\psi\colon C \vdash D$ are combinatorial flows then $\phi[a/\psi]\colon \Gamma[a/C, \bar{a}/\bar{D}] \vdash \Delta[a/D, \bar{a}/\bar{C}]$ is a combinatorial flow. This operation is called *substitution*.

The *size* of a combinatorial flow ϕ, denoted by $|\phi|$, is defined to be the sum of the sizes of all simple combinatorial flows occurring in ϕ.

Theorem 3.9. *Combinatorial flows form a proof system.*

Proof. This immediately follows from Definition 3.8 and Theorem 3.2. □

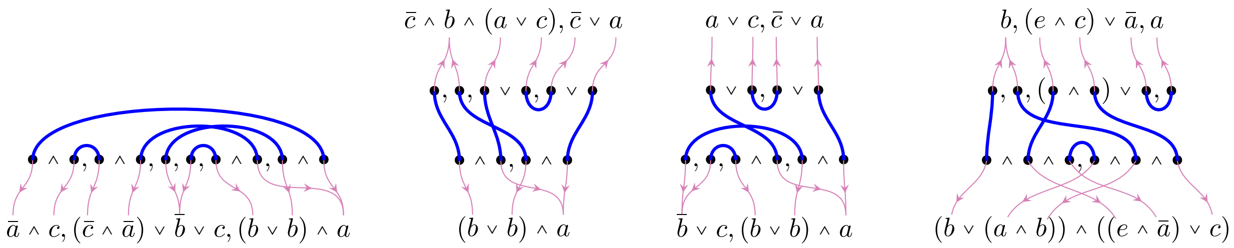
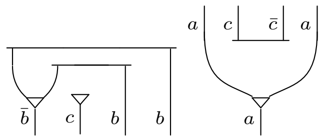


Figure 1. Examples of simple combinatorial flows

Remark 3.10. Theorems 3.2 and 3.9 provide the main advantage of combinatorial flows over B-nets and N-nets [LS05] and atomic flows [GG08], [GGS10]. For a simple combinatorial flow $\phi\colon \circ \vdash \Gamma$, we can immediately obtain the corresponding N-net by forgetting the cograph $\langle V, R\rangle$ and connecting the atoms of Γ according to the (undirected) paths given by f and B. The example below is obtained from the first flow in Figure 1:

$$\bar{a} \wedge c, (\bar{c} \wedge \bar{a}) \vee \bar{b} \vee c, (b \vee b) \wedge a \tag{9}$$

The corresponding B-net is obtained by forgetting the multiplicity of the edges. In the example in (9), the B-net is identical to the N-net. For translating a simple combinatorial flow $\phi\colon \Delta \vdash \Gamma$ into an atomic flow, we not only forget the cograph $\langle V, R\rangle$ but also the structure of Γ and the order of the atoms in Γ. We only look at the paths given by f and B and keep track of which atoms are in Γ and which ones are in Δ. Here is the third example in Figure 1 translated into an atomic flow:



A substitution-free combinatorial flow can straightforwardly translated into atomic flows since they can be composed horizontally and vertically. However, in each translation, critical information is lost, such that it becomes impossible to recover the proof from an N-net or an atomic flow in polynomial time.

Remark 3.11. Another related concept is the notion of *logical flow graph* [Bus91], [Car97], in which all subformulas in all sequents in the proof are vertices, connected by edges determined by the premise/conclusion relation of the inference rules. Comparing to this, in combinatorial flows only the atoms, and only in the endsequent and one particular intermediate sequent are vertices. This "intermediate sequent" is not explicitly visible in the sequent calculus, but it is in the calculus of structures [GS01], [BT01], [Gug07]. This will become evident in Section 8, in particular, Theorem 8.2.

Definition 3.12. A combinatorial flow is *normal* if it is a simple combinatorial flow. It is *cut-free* if the composition operation $\blacklozenge$ is not used in it, and it is *substitution-free* if the substitution operation is not used in it.

Normalization of a combinatorial flow means therefore to remove the operations defined in Definition 3.8. The following four Sections are dedicated to this.

4. Normalization I: Binary Connectives

Lemma 4.1. *Let $\phi\colon A_1 \vdash A_2$ and $\psi\colon B_1 \vdash B_2$ be simple combinatorial flows. Then there are simple combinatorial flows $\chi\colon A_1 \wedge B_1 \vdash A_2 \wedge B_2$ and $\xi\colon A_1 \vee B_1 \vdash A_2 \vee B_2$, such that $|\chi| \leqslant |\phi| + |\psi|$ and $|\xi| \leqslant |\phi| + |\psi|$.*

Proof. Let $\mathfrak{C}$ and $\mathfrak{D}$ be the R&B-cographs for ϕ and ψ, respectively, and let $f\colon \mathfrak{C}^{\downarrow} \rightarrowtail \mathfrak{G}(\bar{A}_1) \vee \mathfrak{G}(A_2)$ and $g\colon \mathfrak{D}^{\downarrow} \rightarrowtail \mathfrak{G}(\bar{B}_1) \vee \mathfrak{G}(B_2)$ be their defining skew fibrations. Then, let $\mathfrak{C}_1$ and $\mathfrak{C}_2$ be the subgraphs of $\mathfrak{C}^{\downarrow}$, and $f_1\colon \mathfrak{C}_1 \to \mathfrak{G}(\bar{A}_1)$ and $f_2\colon \mathfrak{C}_2 \to \mathfrak{G}(A_2)$ be the corresponding restrictions of f, obtained via Lemma 3.3. Similarly, let $\mathfrak{D}_1$ and $\mathfrak{D}_2$ be the corresponding subgraphs of $\mathfrak{D}^{\downarrow}$, and g_1 and g_2 the corresponding restrictions of g.

The simple flow $\chi\colon A_1 \wedge B_1 \vdash A_2 \wedge B_2$ can now be given by the R&B-cograph $\mathfrak{H}$ and skew fibration $h\colon \mathfrak{H}^{\downarrow} \rightarrowtail \mathfrak{G}(\overline{A_1 \wedge B_1}, A_2 \wedge B_2)$ which are defined as follows:

- If $\mathfrak{C}_2$ and $\mathfrak{D}_2$ are both not empty, then we define $\mathfrak{H}^{\downarrow} = \mathfrak{D}_1 \vee \mathfrak{C}_1 \vee (\mathfrak{C}_2 \wedge \mathfrak{D}_2)$, and $B_{\mathfrak{H}} = B_{\mathfrak{C}} \uplus B_{\mathfrak{D}}$, and $h = g_1 \vee f_1 \vee (f_2 \wedge g_2)$. To see that this is well-defined, note that $\mathfrak{G}(\overline{A_1 \wedge B_1}, A_2 \wedge B_2)$ is the same as $\mathfrak{G}(\bar{B}_1) \vee \mathfrak{G}(\bar{A}_1) \vee (\mathfrak{G}(A_2) \wedge \mathfrak{G}(B_2))$.
- If $\mathfrak{C}_2$ is empty then $\mathfrak{C}_1 = \mathfrak{C}^{\downarrow}$ and we define $\mathfrak{H} = \mathfrak{C}$ and let h behave as f does.
- If $\mathfrak{D}_2$ is empty and $\mathfrak{C}_2$ is not, then then $\mathfrak{D}_1 = \mathfrak{D}^{\downarrow}$ and we define $\mathfrak{H} = \mathfrak{D}$ and let h behave as g does.

Then, $\mathfrak{H}$ is an R&B-cograph (by construction) and it is critically chorded. In the first case the situation is the same is in the $\otimes$-rule for MLL-proof nets (see [Ret03]) and in the other two cases it is trivial. It also trivially follows that h is axiom preserving. Therefore it only remains to show that h is indeed a skew fibration. For this, observe that $g_1 \vee f_1 \vee (f_2 \wedge g_2)$ fails to be a skew fibration only if one of $\mathfrak{C}_2$ or $\mathfrak{D}_2$ is empty. On the other hand, f is a skew-fibration from $\mathfrak{C}^{\downarrow}$ to $\mathfrak{G}(\bar{B}_1) \vee \mathfrak{G}(\bar{A}_1) \vee (\mathfrak{G}(A_2) \wedge \mathfrak{G}(B_2))$ if no vertex of $\mathfrak{C}$ is mapped to $\mathfrak{G}(A_2)$, i.e., $\mathfrak{C}_2$ is empty.

Dually, we can define the simple flow $\xi\colon A_1 \vee B_1 \vdash A_2 \vee B_2$. □

Remark 4.2. Note that it is crucial to check whether $\mathfrak{C}_2$ or $\mathfrak{D}_2$ are empty, whereas for $\mathfrak{C}_1$ and $\mathfrak{D}_1$, this is irrelevant. The difference is shown in Figure 2 (and dually in Figure 3). Note also that there is an arbitrary choice to make when both $\mathfrak{C}_2$ and $\mathfrak{D}_2$ are empty.

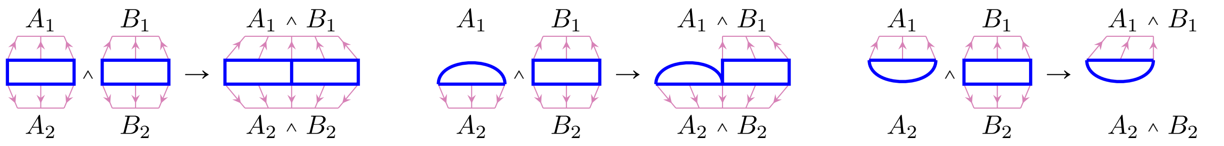


Figure 2. Conjunction of simple combinatorial flows

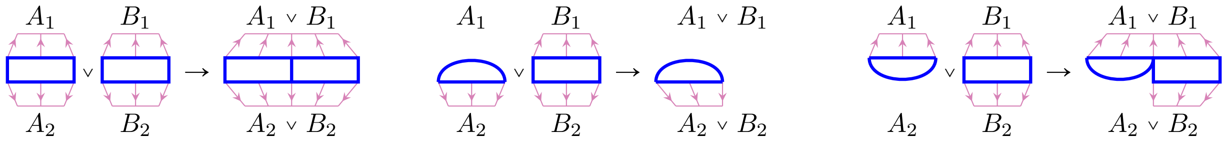


Figure 3. Disjunction of simple combinatorial flows

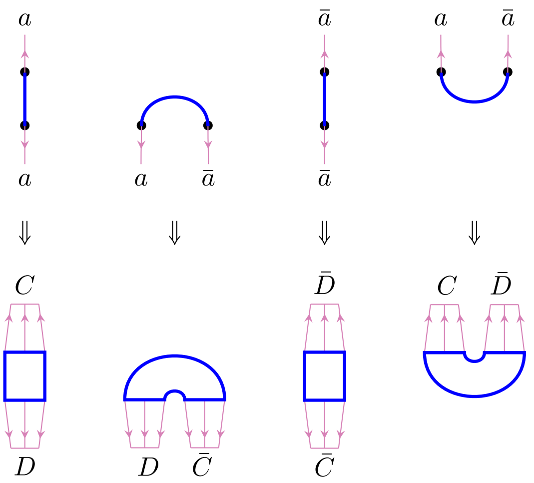
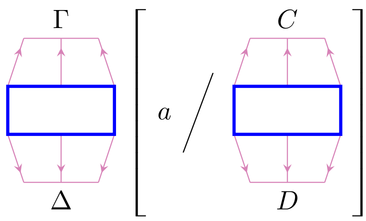


Figure 4. Substitution of simple combinatorial flows

5. Normalization II: Substitution

In this section we eliminate the substitution in the flow $\phi[a/\psi]\colon \Gamma\big[a/C,\bar{a}/\bar{D}\big] \vdash \Delta\big[a/D,\bar{a}/\bar{C}\big]$ shown below



(10)

Lemma 5.1. *Let $\phi\colon \Gamma \vdash \Delta$ and $\psi\colon C \vdash D$ be simple combinatorial flows. Then there is a simple combinatorial flow $\phi'\colon \Gamma[a/C,\bar{a}/\bar{D}] \vdash \Delta[a/D,\bar{a}/\bar{C}]$.*

The basic idea if the construction is as follows: The simple combinatorial flow $\phi\colon \Gamma \vdash \Delta$ consists of simple paths, and each simple path in ϕ whose endpoints are occurrences of a or $\bar{a}$ are replaced according to Figure 4. To define this more formally, we first need the notion of substitution in a graph.

Construction 5.2. Let $\mathfrak{C}$ and $\mathfrak{D}$ be disjoint graphs, and let x be a vertex in $\mathfrak{C}$. With $\mathfrak{C}[x/\mathfrak{D}]$ we denote the graph whose vertex set is $V = V_{\mathfrak{C}}\backslash\{x\} \cup V_{\mathfrak{D}}$ and whose edge set is $E = E_{\mathfrak{C}}\backslash\{xz \mid z \in V_{\mathfrak{C}}\} \cup \{yz \mid y \in V_{\mathfrak{D}}, xz \in E_{\mathfrak{C}}\}$.

In other words, we remove x from $\mathfrak{C}$ and replace it by $\mathfrak{D}$, such that we have an edge from a remaining vertex y in $\mathfrak{C}$ to all vertices in $\mathfrak{D}$, whenever there was an edge from y to x in $\mathfrak{C}$ before.

Lemma 5.3. *If $\mathfrak{C}$ and $\mathfrak{D}$ are cographs and $x \in V_{\mathfrak{C}}$, then $\mathfrak{C}[x/\mathfrak{D}]$ is also a cograph.*

Proof. If we take the formula tree for $\mathfrak{C}$, remove the leaf x, and replace it by the formula tree of $\mathfrak{D}$, we obtain a formula tree for $\mathfrak{C}[x/\mathfrak{D}]$, which is therefore a cograph by Proposition 2.6. □

Construction 5.4. In Construction 5.2 we substituted graphs for *vertexes* in other graphs. Now we use this to substitute R&B-graphs for *B-edges* in other R&B-graphs. Let $\mathfrak{C}$ and $\mathfrak{D}$ be disjoint R&B-graphs, and let $x, y \in V_{\mathfrak{C}}$ with $xy \in B_{\mathfrak{C}}$. Furthermore, let $\mathfrak{D}^{\downarrow} = \mathfrak{D}_1 \vee \mathfrak{D}_2$. We now define the R&B-graph $\mathfrak{H} = \mathfrak{C}[xy/\langle\mathfrak{D}_1 \vee \mathfrak{D}_2, B_{\mathfrak{D}}\rangle] = \langle V_{\mathfrak{H}}, R_{\mathfrak{H}}, B_{\mathfrak{H}}\rangle$ as follows. We let $\langle V_{\mathfrak{H}}, R_{\mathfrak{H}}\rangle = \mathfrak{C}^{\downarrow}[x/\mathfrak{D}_1][y/\mathfrak{D}_2]$, applying Construction 5.2 twice, and let $B_{\mathfrak{H}} = B_{\mathfrak{C}}\backslash\{xy\} \cup B_{\mathfrak{D}}$. In other words, x is replaced by $\mathfrak{D}_1$ and y by $\mathfrak{D}_2$, and the B-edge xy is removed an replaced by the matching $B_{\mathfrak{D}}$.

Lemma 5.5. *If $\mathfrak{C}$ and $\mathfrak{D}$ are R&B-cographs with $xy \in B_{\mathfrak{C}}$ and $\mathfrak{D}^{\downarrow} = \mathfrak{D}_1 \vee \mathfrak{D}_2$ then $\mathfrak{H} = \mathfrak{C}[xy/\langle\mathfrak{D}_1 \vee \mathfrak{D}_2, B_{\mathfrak{D}}\rangle]$ also is an R&B-cograph. Furthermore, if $\mathfrak{C}$ and $\mathfrak{D}$ are both critically chorded, then so is $\mathfrak{H}$.*

Proof. The graph $\mathfrak{H}$ is a cograph for the same reason as in Lemma 5.3. Now assume by way of contradiction that $\mathfrak{H}$ is not critically chorded. First, assume there is a chordless æ-cycle $\mathcal{C}$. If all vertices of $\mathcal{C}$ are inside $V_{\mathfrak{C}}$ or all inside $V_{\mathfrak{D}}$, we have immediately a contradiction to $\mathfrak{C}$ and $\mathfrak{D}$ having no chordless æ-cycle. So, the cycle $\mathcal{C}$ must contain vertices from $V_{\mathfrak{C}}$ and $V_{\mathfrak{D}}$. Since by construction all B-edges are fully contained in $\mathfrak{C}$ or in $\mathfrak{D}$, we must have an R-edge participating in $\mathcal{C}$ and connecting a vertex $u \in V_{\mathfrak{C}}$ to a vertex $z \in V_{\mathfrak{D}}$. Let $v \in V_{\mathfrak{C}}$ be the unique vertex with $uv \in B_{\mathfrak{C}}$. However, since $uz \in R_{\mathfrak{H}}$, we must by construction also have $vz \in R_{\mathfrak{H}}$ which is a chord for $\mathcal{C}$. Contradiction. For showing that that any two vertices in $\mathfrak{H}$ are connected by a chordless path, we can proceed similarly. □

Proof of Lemma 5.1. Let ϕ and ψ as above and let $\Gamma' = \Gamma\big[a/C,\bar{a}/\bar{D}\big]$ and $\Delta' = \Delta\big[a/D,\bar{a}/\bar{C}\big]$. For constructing the simple flow $\phi'\colon \Gamma' \vdash \Delta'$, let $\mathfrak{C}$ and $\mathfrak{D}$ be the R&B-cographs for ϕ and ψ, respectively, and let

$f\colon \mathfrak{C}^{\downarrow} \rightarrowtail \mathfrak{G}(\bar{\Gamma}, \Delta)$ and $g\colon \mathfrak{D}^{\downarrow} \rightarrowtail \mathfrak{G}(\bar{C}, D)$ be their corresponding skew fibrations. For brevity, we write $\mathfrak{G}$ for $\mathfrak{G}(\bar{\Gamma}, \Delta)$, and $\mathfrak{G}'$ for $\mathfrak{G}(\bar{\Gamma}', \Delta')$. Next, let $\mathfrak{D}_{\bar{C}}$ and $\mathfrak{D}_D$ be the two cographs obtained from $\mathfrak{D}^{\downarrow}$ via Lemma 3.3, and let $x_1, \ldots, x_n \in V_{\mathfrak{C}}$ be the vertexes that f maps to a vertex labeled $\bar{a}$ in $\mathfrak{G}$, and and let $y_1, \ldots, y_n \in V_{\mathfrak{C}}$ be all the vertexes that f maps to a vertex labeled a in $\mathfrak{G}$ — their number has to be identical, otherwise f could not be axiom preserving. Without loss of generality, we can assume that $\{x_1y_1, \ldots, x_ny_n\} \subseteq B_{\mathfrak{C}}$. We can now give the R&B-cograph $\mathfrak{C}'$ for ϕ' as follows:

$$\mathfrak{C}' = \mathfrak{C}[x_1y_1/\langle \mathfrak{D}_{\bar{C}} \vee \mathfrak{D}_D, B_{\mathfrak{D}}\rangle] \cdots [x_ny_n/\langle \mathfrak{D}_{\bar{C}} \vee \mathfrak{D}_D, B_{\mathfrak{D}}\rangle]$$

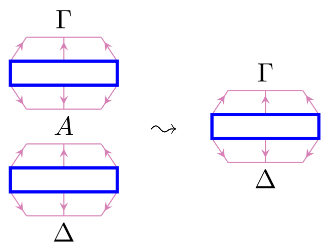
applying Construction 5.4 for each B-edge in $\mathfrak{C}$ connecting an a and an $\bar{a}$ in $\mathfrak{G}$. Finally, we define the map $f'\colon \mathfrak{C}' \rightarrowtail \mathfrak{G}'$ as follows: For every $z \in V_{\mathfrak{C}} \backslash \{x_1, \ldots, x_n, y_1, \ldots, y_n\}$, we have $f'(z) = f(z)$. For each x_i that is mapped by f to a $\bar{a}$, we use g to map the substituted copy of $\mathfrak{D}_{\bar{C}}$ in $\mathfrak{C}'$ to the corresponding substituted copy of $\mathfrak{G}(\bar{C})$ in $\mathfrak{G}'$. We proceed similarly for each y_i. It is easy to see that the so defined f' is indeed a skew fibration and axiom preserving. □

6. Normalization III: Cut

In this section, we show how cuts are eliminated.

Lemma 6.1. *Let $\phi\colon \Gamma \vdash A$ and $\psi\colon A \vdash \Delta$ be simple combinatorial flows. Then there is a simple combinatorial flow $\chi\colon \Gamma \vdash \Delta$.*

Before we give the construction of χ, as indicated below:



(11)

we need first to establish some preliminary properties on skew fibrations and the composition of R&B-cographs.

Lemma 6.2. *Let $\mathfrak{C}$, $\mathfrak{D}$, $\mathfrak{G}$, $\mathfrak{H}$ be cographs.*

1) *If $f\colon \mathfrak{C} \rightarrowtail \mathfrak{G}$ is an isomorphism, then it is also a skew fibration.*
2) *The map $\mathsf{w}\colon \mathfrak{C} \rightarrowtail \mathfrak{C} \vee \mathfrak{D}$, which behaves like the identity on $\mathfrak{C}$, is a skew fibration.*
3) *The map $\mathsf{c}\colon \mathfrak{C} \vee \mathfrak{C} \rightarrowtail \mathfrak{C}$, which maps both copies of $\mathfrak{C}$ in the domain like the identity to the $\mathfrak{C}$ in the codomain, is a skew fibration.*
4) *The map $\mathsf{m}\colon (\mathfrak{C} \wedge \mathfrak{D}) \vee (\mathfrak{G} \wedge \mathfrak{H}) \rightarrowtail (\mathfrak{C} \vee \mathfrak{G}) \wedge (\mathfrak{D} \vee \mathfrak{H})$, which maps each $\mathfrak{C}$, $\mathfrak{D}$, $\mathfrak{G}$, $\mathfrak{H}$ identically to itself, is a skew fibration.*
5) *If $f\colon \mathfrak{C} \rightarrowtail \mathfrak{G}$ and $g\colon \mathfrak{D} \rightarrowtail \mathfrak{H}$ are skew fibrations, then so are $f \vee g\colon \mathfrak{C} \vee \mathfrak{D} \rightarrowtail \mathfrak{G} \vee \mathfrak{H}$ and $f \wedge g\colon \mathfrak{C} \wedge \mathfrak{D} \rightarrowtail \mathfrak{G} \wedge \mathfrak{H}$.*
6) *If $f\colon \mathfrak{C} \rightarrowtail \mathfrak{G}$ and $g\colon \mathfrak{G} \rightarrowtail \mathfrak{H}$ are skew fibrations, then so is $g \circ f\colon \mathfrak{C} \rightarrowtail \mathfrak{H}$.*

Proof. Straightforward. □

Construction 6.3. Let $\mathfrak{C}$ and $\mathfrak{D}$ be R&B-cographs such that $\mathfrak{C}^{\downarrow} = \mathfrak{G} \vee \mathfrak{H}$ and $\mathfrak{D}^{\downarrow} = \bar{\mathfrak{H}} \vee \mathfrak{K}$ for some cographs $\mathfrak{G}$, $\mathfrak{H}$, and $\mathfrak{K}$. We define the graph $\mathfrak{B} = \langle V_{\mathfrak{B}}, E_{\mathfrak{B}}\rangle$ with $V_{\mathfrak{B}} = V_{\mathfrak{G}} \uplus V_{\mathfrak{H}} \uplus V_{\mathfrak{K}}$ and $E_{\mathfrak{B}} = B_{\mathfrak{C}} \uplus B_{\mathfrak{D}}$. This allows us to define the R&B-cograph $\mathfrak{E} = \mathfrak{C} \bullet \mathfrak{D}$ as follows: We let $\mathfrak{E}^{\downarrow} = \mathfrak{G} \vee \mathfrak{K}$, i.e., $V_{\mathfrak{E}} = V_{\mathfrak{G}} \cup V_{\mathfrak{K}}$ and $R_{\mathfrak{E}} = E_{\mathfrak{G}} \cup E_{\mathfrak{K}}$, and we let $xy \in B_{\mathfrak{E}}$ iff there is a path from x to y in $\mathfrak{B}$. Note that this indeed defines a perfect matching. For each x in $V_{\mathfrak{E}}$ there is a unique y connected to x by a path in $\mathfrak{B}$ because $B_{\mathfrak{C}}$ and $B_{\mathfrak{D}}$ are both perfect matchings.

Lemma 6.4. *If in Construction 6.3 the R&B-cographs $\mathfrak{C}$ and $\mathfrak{D}$ are critically chorded, then so is $\mathfrak{E} = \mathfrak{C} \bullet \mathfrak{D}$.*

Proof. This follows directly from the correspondence to MLL^{-} proof nets given in [Ret03] and the standard cut elimination result for linear logic proof nets. The idea used here goes back to [KM71], and a more recent presentation can be found in [Hug05]. □

Next, we define for a simple flow $\phi\colon \Gamma \vdash B \wedge C$ the two *projections* $\phi_l\colon \Gamma \vdash B$ and $\phi_r\colon \Gamma \vdash C$ that are simple flows that "forget" the information about the deleted subformula. Their existence should not be surprising since from a proof of $B \wedge C$ one should be able to recover proofs of B and of C from the same premises.[2]

Construction 6.5. Let $\phi\colon \Gamma \vdash B \wedge C$ be given by the R&B-cograph $\mathfrak{C}$ and the skew fibration $f\colon \mathfrak{C}^{\downarrow} \rightarrowtail \mathfrak{G}(\bigwedge\bar{\Gamma}) \vee (\mathfrak{G}(B) \wedge \mathfrak{G}(C))$. Let $U_C \subseteq V_{\mathfrak{C}}$ be the set of all vertices in $\mathfrak{C}$ that are mapped by f to atom occurrences in C, and let $U_C^{\perp} \subseteq V_{\mathfrak{C}}$ be the smallest set such that

- If $x \in U_C$ and $xy \in B_{\mathfrak{C}}$ and $y \notin U_C$ then $y \in U_C^{\perp}$.
- If $x \in U_C^{\perp}$ and $xy \in B_{\mathfrak{C}}$ and $y \notin U_C$ then $y \in U_C^{\perp}$.
- If $V', V'' \subseteq V_{\mathfrak{C}}$ induce subcographs and $V' \subseteq U_C^{\perp}$ and $V' \cap V'' = \varnothing$ and $V' \cup V''$ induces a subcograph such that for all $v' \in V'$ and $v'' \in V''$ we have $v'v'' \in R_{\mathfrak{C}}$, then also $V'' \subseteq U_C^{\perp}$.[3]

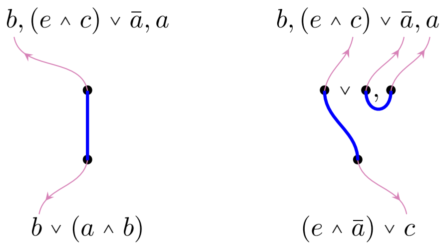
Now let $V_{\mathfrak{C}_l} = V \backslash (U_C \cup U_C^{\perp})$, and let $R_{\mathfrak{C}_l}$ and $B_{\mathfrak{C}_l}$ be the restrictions of $R_{\mathfrak{C}}$ and $B_{\mathfrak{C}}$ (respectively) to $V_{\mathfrak{C}_l}$. Finally, we can define $\phi_l\colon \Gamma \vdash B$ by $\mathfrak{C}_l = \langle V_{\mathfrak{C}_l}, R_{\mathfrak{C}_l}, B_{\mathfrak{C}_l}\rangle$ and $f_l\colon \mathfrak{C}_l^{\downarrow} \rightarrowtail \mathfrak{G}(\bigwedge\bar{\Gamma}) \vee \mathfrak{G}(B)$ which is f restricted to $V_{\mathfrak{C}_l}$.

It is easy to see that $\mathfrak{C}_l$ is critically chorded: any chordless æ-cycle would already be present in $\mathfrak{C}$, and any two vertices are connected by the same chordless æ-path as in $\mathfrak{C}$. We also have that $V_{\mathfrak{C}_l} \neq \varnothing$ (otherwise there would be a chordless æ-cycle in $\mathfrak{C}$.). Finally, it is easy to see that f_l is axiom preserving and a skew fibration. Thus, $\phi_l\colon \Gamma \vdash B$ is indeed a simple combinatorial flow. In the same way we can define the right projection $\phi_r\colon \Gamma \vdash C$. The two examples below show the two projections for the

2. Note, however, that our construction here is different from the one in [Hug06b], due to the absence of mix. Furthermore, we do not need the notion of "laxness".

3. This step can be seen as a combination of the ↓-, ⇓-, and ↑-steps in the empire construction in [BvdW95].

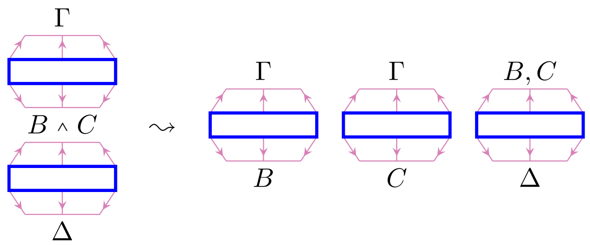
right-most flow in Figure 1:



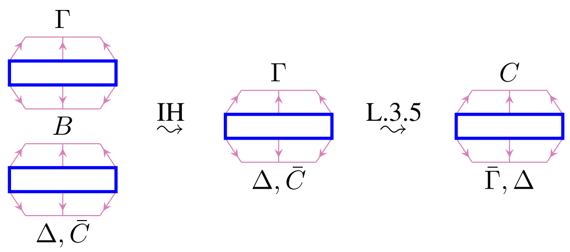
(12)

In a dual way, we can define for a simple combinatorial flow $\psi\colon B \vee C \vdash \Delta$ its left and right projections $\psi_l\colon B \vdash \Delta$ and $\psi_r\colon C \vdash \Delta$.

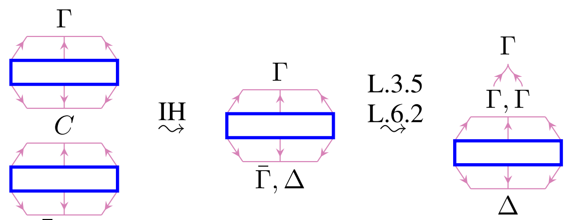
Proof of Lemma 6.1. We proceed by induction on the formula A. First, assume $A = B \wedge C$. Then, from $\phi\colon \Gamma \vdash B \wedge C$ we can obtain the two projections $\phi_l\colon \Gamma \vdash B$ and $\phi_r\colon \Gamma \vdash C$, and from $\psi\colon B \wedge C \vdash \Delta$, we get $\psi'\colon B, C \vdash \Delta$:



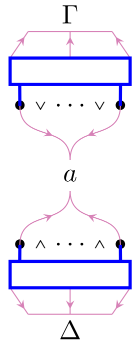
From ψ' we can obtain (via Lemma 3.5) $\psi''\colon B \vdash \Delta, \bar{C}$, which can be composed with ϕ_l to get, by induction hypothesis, a simple flow $\xi\colon \Gamma \vdash \Delta, \bar{C}$, from which (again by Lemma 3.5) we can get a simple flow $\chi'\colon C \vdash \bar{\Gamma}, \Delta$:



This can be composed with ϕ_r, which gives us by induction hypothesis a simple flow $\chi''\colon \Gamma \vdash \bar{\Gamma}, \Delta$, from which we get a simple flow $\chi'\colon \Gamma, \Gamma \vdash \Delta$ by applying Lemma 3.5. Finally, we can apply Lemma 6.2 to get the desired $\chi\colon \Gamma \vdash \Delta$:



If $A = B \vee C$ we proceed analogous. It remains to show the case when A is an atom, i.e., we have the situation:

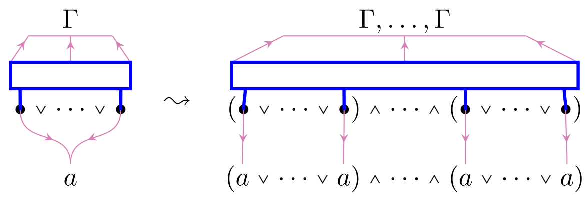


(13)

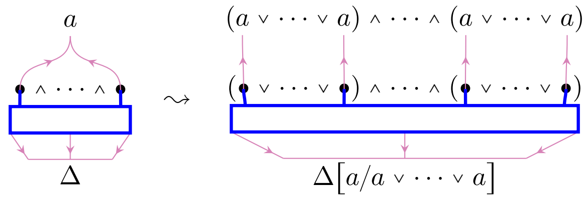
Let $f\colon \mathfrak{C}^{\downarrow} \rightarrowtail \mathfrak{G}(\bigwedge\bar{\Gamma}, a)$ and $g\colon \mathfrak{D}^{\downarrow} \rightarrowtail \mathfrak{G}(\bar{a}, \bigvee\Delta)$ be the skew fibrations of the simple flows $\phi\colon \Gamma \vdash a$ and $\psi\colon a \vdash \Delta$, respectively. Let $x_1, \ldots, x_n$ be the vertices in $\mathfrak{C}$ that are mapped by f to the a in the conclusion of ϕ, and let $y_1, \ldots, y_m$ be the vertices in $\mathfrak{D}$ that are mapped by g to the occurrence of $\bar{a}$ that represents the a in the premise of ψ.

Now we define the map $f^*\colon \mathfrak{C}^{\downarrow} \rightarrowtail \mathfrak{G}(\bigwedge\bar{\Gamma}, a \vee \cdots \vee a)$ where we replace a by a disjunction of n copies of a, and let f^* behave as f on $V_{\mathfrak{C}} \setminus \{x_1, \ldots, x_n\}$ and map each x_i to one copy of a. This clearly also is a skew fibration, and in a similar way we define the skew fibration $g^*\colon \mathfrak{D}^{\downarrow} \rightarrowtail \mathfrak{G}(\bar{a} \vee \cdots \vee \bar{a}, \bigvee\Delta)$ where we use m copies of $\bar{a}$. We let $\phi^*\colon \Gamma \vdash a \vee \cdots \vee a$ and $\psi^*\colon a \wedge \cdots \wedge a \vdash \Delta$ be the simple flows defined by f^* and g^*, respectively.

We now apply the construction of Section 4 to form the conjunction of m copies of ϕ^*, which yields a simple flow $\hat{\phi}\colon \Gamma, \ldots, \Gamma \vdash (a \vee \cdots \vee a) \wedge \cdots \wedge (a \vee \cdots \vee a)$, as indicated below:



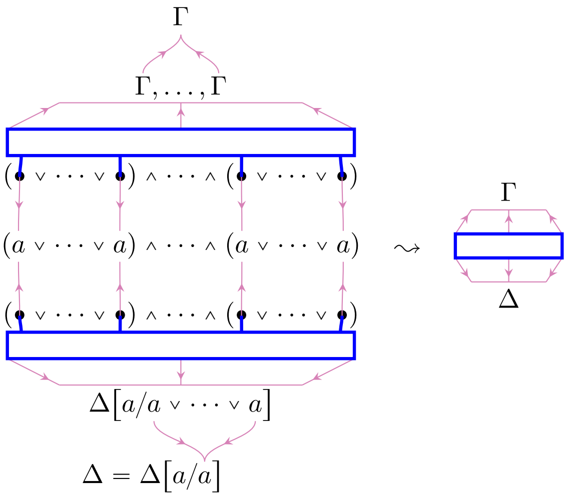
Next, we substitute in ψ^* all simple flow paths that start in the premise $a \wedge \cdots \wedge a$ by the identity flow $\mathsf{id}\colon a \vee \cdots \vee a \vdash a \vee \cdots \vee a$ (with m copies of a on each side) as done in Section 5. Then we have a simple flow $\hat{\psi}\colon (a \vee \cdots \vee a) \wedge \cdots \wedge (a \vee \cdots \vee a) \vdash \Delta[a/a \vee \cdots \vee a]$ as shown below:[4]



We now plug $\hat{\phi}$ and $\hat{\psi}$ together and apply Lemma 6.4 to get a simple flow $\chi'\colon \Gamma, \ldots, \Gamma \vdash \Delta[a/a \vee \cdots \vee a]$, to

4. There is a slight abuse of notation: $\Delta[a/a \vee \cdots \vee a]$ stands for the sequent obtained by replacing every occurrence of a in Δ from which there is a simple flow path to an a in the premise of ψ^* by $a \vee \cdots \vee a$ (i.e., there might be occurrences of a in Δ that are not replaced).

which we apply Lemma 6.2



to get the desired simple flow $\chi\colon \Gamma \vdash \Delta$. □

Remark 6.6. The non-confluence of cut elimination in classical logic does not disappear by this method. There is a non-deterministic choice in the atomic case. Either we duplicate ϕ and perform a substitution in ψ, as we did here, or we do the substitution $a/a \wedge \cdots \wedge a$ (with m copies of a) in ϕ and make n copies of ψ.

7. Normalization IV: Putting things together

If we define the relation $\to$ on combinatorial flows such that $\phi_1 \to \phi_2$ whenever ϕ_1 can be reduced to ϕ_2 by one of the reductions given by Lemmas 4.1, 5.1, and 6.1, then we have immediately the following:

Theorem 7.1. *The relation $\to$ is strongly normalizing, and the normal forms are the simple combinatorial flows.*

Proof. At each step the number of simple combinatorial flows in the flow is reduced, and we always can make at least one reduction when the flow is not simple. □

Corollary 7.2. *For each combinatorial flow $\phi\colon \Gamma \vdash \Delta$ there is a simple combinatorial flow $\phi'\colon \Gamma \vdash \Delta$ with the same premise and conclusion.*

Note that the reduction $\to$ reduces horizontal composition, cuts, ans substitutions at the same time, proceeding according to the term structure of the given combinatorial flow. As mentioned before the reduction step eliminating horizontal composition (Section 4) does not increase the size of the flow. However, cut elimination (Section 6) as well as substitution elimination (Section 5) can lead to an exponential blow-up. Thus, both cut and substitution can be seen as mechanisms to compress the flow.

8. Relation to to deep inference proofs

In the remainder of this paper we show how combinatorial flows are related to syntactic proofs given in some deductive formalism. We start with proofs in the deep inference system SKS [BT01], which is shown in Figure 5.[5] The rules shown there should be read as rewrite rule schemes that can be applied inside an arbitrary (positive) formula context. In the rules ai↓, ai↑, ac↓, and ac↑, the a can stand for any atom. In all rules, A, B, C, and D, can stand for any formula, and in ai↓ we additionally allow A to be empty, so to have proper proofs without premise.[6] We write

$$
\mathsf{ai}{\downarrow}\,\frac{A}{A \wedge (a \vee \bar{a})} \qquad \mathsf{s}\,\frac{(A \vee B) \wedge C}{A \vee (B \wedge C)} \qquad \mathsf{ai}{\uparrow}\,\frac{(\bar{a} \wedge a) \vee A}{A}
$$

$$
\mathsf{ac}{\downarrow}\,\frac{a \vee a}{a} \qquad \mathsf{m}\,\frac{(A \wedge C) \vee (B \wedge D)}{(A \vee B) \wedge (C \vee D)} \qquad \mathsf{ac}{\uparrow}\,\frac{a}{a \wedge a}
$$

$$
\mathsf{w}{\downarrow}\,\frac{A}{A \vee B} \qquad \mathsf{w}{\uparrow}\,\frac{B \wedge A}{A}
$$

Figure 5. Deep inference system SKS

$$
\begin{array}{c} P \\ \mathsf{S}\,\Vert\,\Phi \\ Q \end{array} \quad \text{and} \quad \begin{array}{c} \mathsf{S}\,\Vert\,\Psi \\ Q \end{array} \tag{14}
$$

to denote that there is a derivation Φ from P to Q, (respectively the proof Ψ without premise for the formula Q) in the system S, modulo the equivalence relation defined by associativity and commutativity of $\wedge$ and $\vee$, as given in (4). Below is an example of a derivation in SKS

$$
\begin{array}{rc}
\mathsf{aw}{\uparrow} & \bar{c} \wedge b \wedge (a \vee c) \wedge (\bar{c} \vee a) \\ \hline
\mathsf{ac}{\uparrow} & b \wedge (a \vee c) \wedge (\bar{c} \vee a) \\ \hline
2 \cdot \mathsf{s} & b \wedge b \wedge (a \vee c) \wedge (\bar{c} \vee a) \\ \hline
\mathsf{ai}{\uparrow} & b \wedge b \wedge (a \vee (c \wedge \bar{c}) \vee a) \\ \hline
2 \cdot \mathsf{s} & b \wedge b \wedge (a \vee a) \\ \hline
\mathsf{m} & (b \wedge a) \vee (b \wedge a) \\ \hline
\mathsf{ac}{\downarrow} & (b \vee b) \wedge (a \vee a) \\ \hline
 & (b \vee b) \wedge a
\end{array} \tag{15}
$$

where $2 \cdot \mathsf{s}$ stands for two consecutive applications of the s-rule. The *size* of a derivation Φ, denoted by $|\Phi|$ is the sum of the sizes of the formula occurrences in Φ.

Each rule in system SKS can straightforwardly be translated into a simple combinatorial flow, as indicated in Figure 6, where the double lines indicate the identity (see Observation 3.6). Note that for the m-rule there are two possible translations. Since whenever $A = B$ modulo associativity and commutativity (4) we have that $\mathfrak{G}(A) = \mathfrak{G}(B)$, an equivalence step in an SKS-proof can translated into the identity flow. This is enough to give a direct translation which proves the following:

Theorem 8.1. *Substitution-free combinatorial flows p-simulate system* SKS.

Proof. We show that for every SKS derivation Ψ with premise P and conclusion Q, there is a substitution-free combinatorial flow $\psi\colon P \vdash Q$ whose size is quadratic in $|\Psi|$. For every rule $\mathsf{r}\,\dfrac{A}{B}$ in Figure 5 we have a simple combinatorial flow $\phi_{\mathsf{r}}\colon A \vdash B$, as shown in Figure 6. By

5. Note that there is a slight abuse of notation since our system different from the original version of SKS in [BT01]: We do not have explicit units in the language, and therefore our weakening rule is not atomic (see also [Str12]).

6. We could also allow A to be empty in ai↑, so to have a proper refutation without conclusion.

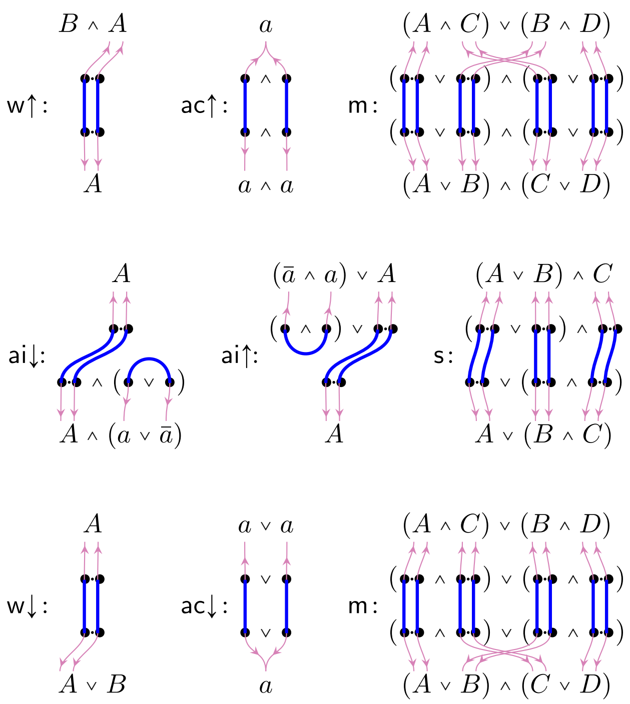


Figure 6. Simple combinatorial flows for the rules in Figure 5

horizontally composing it sufficiently often with the identity flow (8), we can for every (positive) formula context $F\{\ \}$ produce a simple flow $\phi_{F\{\mathsf{r}\}}\colon F\{A\} \vdash F\{B\}$. We now proceed by induction on the length of Ψ to produce ψ:

$$\mathsf{r}\,\frac{\begin{array}{c} P \\ \mathsf{SKS}\big\Vert\Psi' \\ F\{A\} \end{array}}{F\{B\}} \quad\rightsquigarrow\quad \psi = \psi' \bullet \phi_{F\{\mathsf{r}\}}\colon P \vdash F\{B\}$$

where $Q = F\{B\}$ and $\psi'\colon P \vdash F\{A\}$ exists by induction hypothesis. ☐

Before we look at the other direction, we first look at the expressive power of simple combinatorial flows. We have the following result:

Theorem 8.2. *Let A and B be formulas. There is a simple combinatorial flow $\phi\colon A \to B$ iff there are formulas A' and B' such that there are derivations*

$$\begin{array}{c} A \\ {\scriptstyle\{\mathsf{w}\uparrow,\mathsf{ac}\uparrow,\mathsf{m}\}}\big\Vert\Phi_1 \\ A' \end{array} \quad\text{and}\quad \begin{array}{c} A' \\ {\scriptstyle\{\mathsf{ai}\downarrow,\mathsf{ai}\uparrow,\mathsf{s}\}}\big\Vert\Phi_2 \\ B' \end{array} \quad\text{and}\quad \begin{array}{c} B' \\ {\scriptstyle\{\mathsf{w}\downarrow,\mathsf{ac}\downarrow,\mathsf{m}\}}\big\Vert\Phi_3 \\ B \end{array}$$

and such that $|\phi| = O(|\Phi_1|+|\Phi_2|+|\Phi_3|)$. Similarly, there is a simple flow $\psi\colon \circ \vdash B$ iff there are derivations

$$\begin{array}{c} {\scriptstyle\{\mathsf{ai}\downarrow,\mathsf{s}\}}\big\Vert\Psi_2 \\ B' \end{array} \quad\text{and}\quad \begin{array}{c} B' \\ {\scriptstyle\{\mathsf{w}\downarrow,\mathsf{ac}\downarrow,\mathsf{m}\}}\big\Vert\Psi_3 \\ B \end{array}$$

such that $|\psi| = O(|\Psi_2| + |\Psi_3|)$

Proof. First, assume we have an SKS derivation as shown on the left in Figure 7. We let $\mathfrak{C}^{\downarrow} = \mathfrak{G}(\bar{A}') \vee \mathfrak{G}(B')$. By Lemma 8.4 we have a matching $B_{\mathfrak{C}}$ such that the R&B-graph $\mathfrak{C}$ is critically chorded. Then we can form the SKS derivation using only rules $\mathsf{w}\!\downarrow$, $\mathsf{ac}\!\downarrow$ and m from $\bar{A}' \vee B'$ to $\bar{A} \vee B$ by horizontally composing the dual of Φ_1 with Φ_3. By Lemma 8.3 we get our skew fibration $f\colon \mathfrak{C}^{\downarrow} \rightarrowtail \mathfrak{G}(\bar{A}) \vee \mathfrak{G}(B)$. Conversely, let ϕ be given, let $f\colon \mathfrak{C}^{\downarrow} \rightarrowtail \mathfrak{G}(\bar{A}, B)$ be its skew fibration, and let $\mathfrak{C}_{\bar{A}}$ and $\mathfrak{C}_B$ be the two cographs obtained via Lemma 3.3. If we add labels to $\mathfrak{C}_{\bar{A}}$ and $\mathfrak{C}_B$ such that f is label-preserving, we can let A' and B' be the formulas determined by $\overline{\mathfrak{C}_{\bar{A}}}$ and $\mathfrak{C}_B$, respectively. We can now apply Lemma 8.4 to get Φ_2, and Lemma 8.3 to get Φ_3 and (the dual of) Φ_1. ☐

Figure 7 depicts on the left the two statements of this theorem, and shows on the right two examples. The first is the derivation in (15) enriched with the "flow-graph" tracing the atoms in the derivation. The corresponding combinatorial flow is the second example in Figure 1. The right-most example in Figure 7 corresponds to the third simple flow in Figure 1.

The proof of Theorem 8.2 essentially consists of the following two lemmas:

Lemma 8.3. *Let A and B be formulas. There is a skew fibration $f\colon \mathfrak{G}(A) \rightarrowtail \mathfrak{G}(B)$ iff there is a derivation Φ from A to B in $\{\mathsf{w}\!\downarrow, \mathsf{ac}\!\downarrow, \mathsf{m}\}$.*

Proof. First assume Φ is given. Then we can obtain f by composing the maps that are induced by the rule applications in Φ. That this is a skew fibration follows from Lemma 6.2. Conversely, assume f is given. Let us call a vertex in B *good* if it is in the image of f, and otherwise *bad*. Observe that whenever a vertex a in $\mathfrak{G}(B)$ is bad it cannot be connected by an edge to a good vertex. Since there is at least one good vertex, we have for every bad a a subformula $C \vee D$ in B such that (i) a is inside D, (ii) C contains a good vertex, and furthermore (iii) all vertices in D are bad. We can therefore apply $\mathsf{w}\!\downarrow$ deleting the D. Let B_0 be the formula obtained from B by repeating this process until no bad vertices remain. Then, for each atom in a define n_a be the number of vertices in $\mathfrak{G}(A)$ that f maps to a, and let B' be the formula obtained from B_0 by replacing each a by $a \vee \cdots \vee a$ where there are n_a copies. Then there is a derivation from B' to B_0 using only the $\mathsf{ac}\!\downarrow$-rule. We can define the map $f'\colon \mathfrak{G}(A) \rightarrowtail \mathfrak{G}(B')$ which takes each vertex that f maps to a to one for the new copies of a such that f is now a bijection. It is easy to see that f' is still a skew fibration. Now it follows from [Str07, Theorem 5.1] that there is a derivation from A to B' using only m. Alternatively, this can also be shown using [Hug06b, Theorem 3.2] and the fact that a general contraction can be decomposed into $\mathsf{ac}\!\downarrow$ and m [BT01]. ☐

Lemma 8.4. *Let A and B be formulas. There is a critically chorded R&B-graph $\mathfrak{C}$ with $\mathfrak{C}^{\downarrow} = \mathfrak{G}(\bar{A}) \vee \mathfrak{G}(B)$ iff there is a derivation Φ from A to B in $\{\mathsf{ai}\!\downarrow, \mathsf{ai}\!\uparrow, \mathsf{s}\}$.*

Proof. This follows from the equivalence of critically chorded R&B-graphs to MLL^{-} proofs nets [Ret03] and the fact that $\{\mathsf{ai}\!\downarrow, \mathsf{ai}\!\uparrow, \mathsf{s}\}$ is sound and complete for MLL^{-} (shown in, e.g., [Ret93], [Str03b], [Str03a]). A direct translation from Φ into a critically chorded R&B-graph can also be obtained via Lemma 6.4. ☐

The cut-free variant of SKS is called KS, and consists of the rules $\{\mathsf{ai}\!\downarrow, \mathsf{s}, \mathsf{ac}\!\downarrow, \mathsf{aw}\!\downarrow, \mathsf{m}\}$.

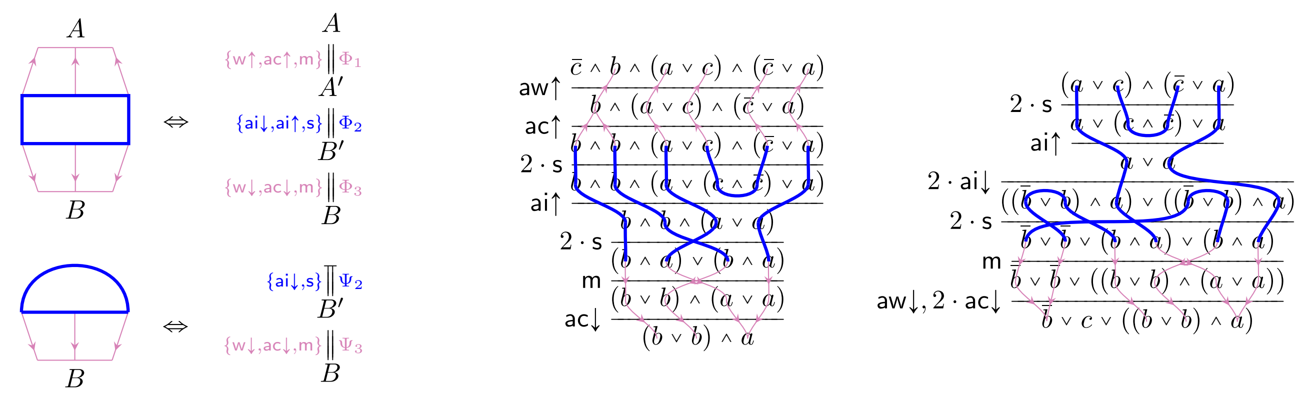


Figure 7. Left: Statements of Theorem 8.2 Right: Examples corresponding to the second and third example in Figure 1

Corollary 8.5. *System* KS *p-simulates simple combinatorial flows.*

Proof. This follows immediately from the second part of Theorem 8.2. ☐

Remark 8.6. Note that the other direction does not follow. In a KS proof, the rules s and m can be interleaved, and this behavior cannot be captured by simple combinatorial flows.

Corollary 8.7. *System* SKS *p-simulates substitution-free combinatorial flows.*

Proof. Let ϕ be a substitution-free combinatorial flow. Then every simple flow occurring in ϕ can be translated into a SKS derivation (as shown on the left in Figure 7). These derivations can be composed horizontally and vertically, according to the structure of ϕ. ☐

After having established the substitution-free combinatorial flows are p-equivalent to SKS, let us now investigate what happens when substitution is present. The substitution rule in a deductive system is given as follows:

$$\mathsf{sub}\,\frac{A}{A\sigma} \tag{16}$$

It replaces a formula A by the formula that is obtained by applying the substitution σ to A.

We define sSKS to be the system SKS + sub. It is important to note that unlike the other rules (shown in Figure 5) the rule sub in (16) cannot be applied inside a context. It is always applied to the whole formula. The reason is that the rule is not "strongly sound", in the sense that the premise does not imply the conclusion, as it is the case with the other inference rules. This means, in particular, that it does not make sense to speak of derivations in sSKS, but only of proofs with no premise.

Theorem 8.8. *Combinatorial flows p-simulate* sSKS.

The basic idea of the proof is to simulate the application of a substitution $\sigma = \{a_1 \mapsto B_1, \ldots, a_n \mapsto B_n\}$ in the sub-rule in sSKS by the substitution of the identity flow id_{B_i} for the variable a_i for each $i = 1..n$. But since in combinatorial flows the replacement is not performed simultaneously, we have to do a renaming first, in order to avoid unwanted variable capturing.

Proof of Theorem 8.8. We show that for every sSKS proof Ψ of a formula Q, there is a combinatorial flow $\psi\colon \circ \vdash Q$ whose size is quadratic in $|\Psi|$. We proceed as in the proof of Theorem 8.1 by induction on the length of Ψ. It only remains to show the case where the bottommost rule applied is sub:

$$\mathsf{sub}\,\frac{\begin{array}{c}\mathsf{sSKS}\,\Big\Vert\,\Psi' \\ A\end{array}}{A\big[a_1/B_1, \bar{a}_1/\bar{B}_1, \ldots, a_n/B_n, \bar{a}_n/\bar{B}_n\big]}$$

where the substitution is $\sigma = \{a_1 \mapsto B_1, \ldots, a_n \mapsto B_n\}$. By induction hypothesis there is a combinatorial flow $\psi'\colon \circ \vdash A$. In ψ' we replace all occurrences of $a_1, \ldots, a_n$ by fresh variables $a'_1, \ldots, a'_n$, respectively (and $\bar{a}_1, \ldots, \bar{a}_n$ by $\bar{a}'_1, \ldots, \bar{a}'_n$). The important point is that $a'_1, \ldots, a'_n$ are chosen such that none of $B_1, \ldots, B_n$ contains any occurrence of one of $a'_1, \bar{a}'_1, \ldots, a'_n, \bar{a}'_n$. Then we can form the combinatorial flow

$$\psi = \psi'[a'_1/\mathsf{id}_{B_1}] \ldots [a'_n/\mathsf{id}_{B_n}]$$

where id_{B_i} is the identity flow for B_i (see Observation 3.6). Then the conclusion of ψ is $A\sigma = A\big[a_1/B_1, \bar{a}_1/\bar{B}_1, \ldots, a_n/B_n, \bar{a}_n/\bar{B}_n\big]$. ☐

For the other direction, some more work is necessary. The reason is that in sSKS, substitution is a global rule, whereas in combinatorial flows it is a local activity, which is more flexible. To solve this problem, we use the notion of *extension*, due to [Tse68], which allows for abbreviations in a syntactic proof, by allowing additional proper axioms (called *extension axioms*) of the shape

$$a_i \Leftrightarrow B_i \qquad\qquad 1 \leqslant i \leqslant k \tag{17}$$

where k is a fixed natural number, the a_i are fresh propositional variables (called *extension variables*) which abbreviate formulas B_i (called *extension formulas*), such that

$$\begin{array}{l}\text{a variable } a_i \text{ does neither occur in the premise}\\ \text{of the proof, nor in the conclusion of the proof,}\\ \text{nor in any } B_j \text{ with } j \leqslant i.\end{array} \tag{18}$$

There are two ways to add extension to SKS. The first, as done in [BG09], is to use the conjunction of the axioms (17) as premise in an SKS-derivation (14). The second one, as done in [Str12], [NS15], is to transform the axioms (17) into deep inference rules:

$$\mathsf{ext}{\downarrow}\,\frac{a_i}{B_i}\qquad \mathsf{ext}{\downarrow}\,\frac{\bar{a}_i}{\bar{B}_i}\qquad \mathsf{ext}{\uparrow}\,\frac{\bar{B}_i}{\bar{a}_i}\qquad \mathsf{ext}{\uparrow}\,\frac{B_i}{a_i}\qquad (19)$$

Unlike the substitution rule (16), the rules in (19) can be applied deeply in any context (provided the global condition (18) is satisfied), and thus are local, and can be used to simulate the substitution in combinatorial flows. We let $\mathsf{eSKS} = \mathsf{SKS} + \mathsf{ext}{\downarrow} + \mathsf{ext}{\uparrow}$.

Theorem 8.9. *System* eSKS *p-simulates combinatorial flows.*

Proof. The basic idea is to use the extension rules in (19) to simulate the substitution in combinatorial flows. We show that for every combinatorial flow $\psi\colon P \vdash Q$ there is a eSKS derivation Ψ with premise P and conclusion Q whose size is polynomial in $|\psi|$. We proceed as in the proof of Corollary 8.7 by induction on the structure of ψ, and we only have to consider the case where $\psi = \phi[a/\pi]\colon A[a/C, \bar{a}/\bar{D}] \vdash B[a/D, \bar{a}/\bar{C}]$ with $\phi\colon A \vdash B$ and $\pi\colon C \vdash D$. We have the following three derivations

$$\begin{array}{c} A \\ \mathsf{eSKS}\big\Vert\Phi \\ B \end{array}\qquad \begin{array}{c} C \\ \mathsf{eSKS}\big\Vert\Pi \\ D \end{array}\qquad \begin{array}{c} \bar{D} \\ \mathsf{eSKS}\big\Vert\bar{\Pi} \\ \bar{C} \end{array}\quad .$$

where the first two exist by induction hypothesis, and the third one is the contrapositive of the second. Let $a_1, \ldots, a_l$ be the extension variables of Φ, let $a_{l+1}, \ldots, a_k$ be the ones of Π, and let $E_1, \ldots, E_k$ be the corresponding extension formulas. By induction hypothesis, Φ and Π both obey (18), and without loss of generality we can assume that none of $a_1, \ldots, a_k$ occurs in any of A, B, C, D. Thus, (18) is also globally satisfied, and this remains so when we add $a \Leftrightarrow C$ as additional extension axiom, i.e., we let $a_{k+1} = a$ and $E_{k+1} = C$. Now we can let Ψ be the following derivation:

$$\begin{array}{c} A[a/C, \bar{a}/\bar{D}] \\ \mathsf{eSKS}\big\Vert\bar{\Pi}^* \\ A[a/C, \bar{a}/\bar{C}] \\ \{\mathsf{ext}{\uparrow}\}\big\Vert\Theta_2 \\ A \\ \mathsf{eSKS}\big\Vert\Psi \\ B \\ \{\mathsf{ext}{\downarrow}\}\big\Vert\Theta_1 \\ B[a/C, \bar{a}/\bar{C}] \\ \mathsf{eSKS}\big\Vert\Pi^* \\ B[a/D, \bar{a}/\bar{C}] \end{array}$$

where Θ_1 and Θ_2 consist only of application of the extension rules (19) for the new axiom $a \Leftrightarrow C$. The derivation Π^* consists of several copies of Π, one for each occurrence of a in B, and similarly, $\bar{\Pi}^*$ consists of several copies of $\bar{\Pi}$, one for each occurrence of $\bar{a}$ in A. □

Corollary 8.10. *Combinatorial flows and* sSKS *and* eSKS *are all p-equivalent to each other.*

Proof. We only need to show that sSKS p-simulates eSKS. So, let Φ be a proof of a formula B in eSKS. We can replace all instances of $\mathsf{ext}{\uparrow}$ in Φ by derivations in the system $\{\mathsf{ai}{\downarrow}, \mathsf{ai}{\uparrow}, \mathsf{s}, \mathsf{ext}{\downarrow}\}$ whose size is polynomial in the size of the used extension formula. (How this is done can already be found in [GS01, Thm.2.6 and Rem.2.8].) Then we can use the construction in [Str12] to obtain the corresponding proof Φ' of B in sSKS. □

$$\mathsf{id}\,\frac{}{a, \bar{a}}\qquad \vee\,\frac{\Gamma, A, B}{\Gamma, A \vee B}\qquad \wedge\,\frac{\Gamma, A \qquad \Delta, B}{\Gamma, \Delta, A \wedge B}$$

$$\mathsf{weak}\,\frac{\Gamma}{\Gamma, A}\qquad \mathsf{cont}\,\frac{\Gamma, A, A}{\Gamma, A}\qquad \mathsf{cut}\,\frac{\Gamma, A \qquad \Delta, \bar{A}}{\Gamma, \Delta}$$

Figure 8. A sequent calculus for classical logic

9. Sequent Calculus and Frege Systems

In [Hug06b], Hughes has already shown how to translate (cut-free) sequent calculus proofs into combinatorial proofs, using the notion of *lax combinatorial proofs*. We show here a translation of sequent proofs into simple combinatorial flows that does not need this detour. Then we show how sequent proofs with cut are translated into (substitution-free) combinatorial flows with cut.

Figure 8 shows the one-sided sequent calculus for classical propositional logic that we will use here, and that we call LK.[7] But it should be clear that any other sound and complete sequent system could be used as well. A proof Π in LK is called *cut-free* if it does not use the cut-rule. The *size* of a proof Π in LK, denoted by $|\Pi|$ is the sum of the sizes of the sequents occurring in Π.

Theorem 9.1. *Simple combinatorial flows p-simulate cut-free* LK.

Proof. We proceed by induction on the structure of Π and make a case analysis on the bottom-most rule application in Π. In the case of id we use the identity flow for a (see Observation 3.6). The case for the $\vee$-rule is trivial since a simple combinatorial flow for Γ, A, B is the same as a combinatorial flow for $\Gamma, A \vee B$. For cont and weak we compose the skew fibration that we have by induction hypothesis with the corresponding skew fibration that is given by Lemma 6.2. The only non-trivial case is the one for $\wedge$. By induction hypothesis we have two simple flows $\pi_1\colon \circ \vdash \Gamma, A$ and $\pi_2\colon \circ \vdash B, \Delta$. By Lemma 3.5 we have two simple flows $\pi_1'\colon \bar{\Gamma} \vdash A$ and $\pi_2\colon \bar{\Delta} \vdash B$. To these, we apply the construction of Section 4 to get the simple flow $\pi'\colon \bar{\Gamma}, \bar{\Delta} \vdash A \wedge B$. We apply again Lemma 3.5 to get $\pi\colon \circ \vdash \Gamma, \Delta, A \wedge B$, as desired. □

Theorem 9.2. *Cut-free* LK *p-simulates simple combinatorial flows.*

Proof (Sketch). Let $\phi\colon \circ \vdash \Gamma$ be a simple flow given by the critically chorded R&B-graph $\mathfrak{C}$ and the skew fibration $f\colon \mathfrak{C}^{\downarrow} \rightarrowtail \mathfrak{G}(\Gamma)$. Then $\mathfrak{C}$ represents an MLL^- proof of the sequent Γ' that is obtained from the formula tree of $\mathfrak{C}$ where the vertices are labeled with their image under f. Therefore we have a sequent proof of Γ' using the rules

7. There is a slight abuse of the name, since Gentzen's original LK [Gen34] was a two-sided system, and the rule for disjunction was "additive". However, the spirit is the same, in the sense that the branching rules (cut and conjunction) are multiplicative, and there are explicit weakening and contraction rules.

id, $\vee$, and $\wedge$ in the first line in Figure 8. Then, it has been shown in [Hug06b], [Str07] that every skew fibration can be obtained from isomorphisms and the maps c and w (see Lemma 6.2) by horizontal and vertical composition. Thus, we can derive Γ from Γ' by "deep applications" of weak and cont. These can now be permuted up until they are applied at the root of the formulas. □

Note that the proof of Theorem 9.1 makes crucial use of Lemma 3.5. Fortunately, we can have a similar result for combinatorial flows:

Lemma 9.3. *Let Γ, Δ, Σ be sequents. For every combinatorial flow $\phi\colon \Sigma, \Gamma \vdash \Delta$ there is a combinatorial flow $\phi^*\colon \Gamma \vdash \bar{\Sigma}, \Delta$, and vice versa.*

Proof. Let $\phi_1 = \mathsf{id}^{\vee}_{\Sigma} \wedge \mathsf{id}_{\Gamma}\colon \bigwedge\Gamma \vdash (\bigvee\bar{\Sigma} \vee \bigwedge\Sigma) \wedge \bigwedge\Gamma$ (see Observation 3.6), let $\phi_2\colon (\bigvee\bar{\Sigma} \vee \bigwedge\Sigma) \wedge \bigwedge\Gamma \vdash \bigvee\bar{\Sigma} \vee (\bigwedge\Sigma \wedge \bigwedge\Gamma)$ be the simple flow corresponding to s in SKS (see Figure 6), and let $\phi_3 = \mathsf{id}_{\bar{\Sigma}} \vee \phi\colon \bigvee\bar{\Sigma} \vee (\bigwedge\Sigma \wedge \bigwedge\Gamma) \vdash \bigvee\bar{\Sigma} \vee \bigvee\Delta$. Then we can let $\phi^* = \phi_1; \phi_2; \phi_3$. The other direction is similar. □

Remark 9.4. Observe that here we do not have a bijection (as it is the case for simple flows), and that the construction is not size preserving as in Lemma 3.5. However, we do have that $|\phi^*| = O(|\phi|)$, which is enough for the following.

Theorem 9.5. *Substitution-free combinatorial flows are p-equivalent to* LK.

Proof (Sketch). The material presented in this paper allows us to give two simple ways to directly translate LK-proofs into combinatorial flows. First, we can use the same construction as for Theorem 9.1, using Lemma 9.3 instead of Lemma 3.5, and use the vertical flow composition to simulate the cut-rule in LK. Second, we can use the same construction as in the proof of Theorem 8.1: for each rule in Figure 8 we can give a corresponding simple flow, which we then compose according to the structure of Π.

Alternatively, we can prove the theorem by using the fact that LK and SKS are p-equivalent [BT01], [BG09] and conclude by Theorem 8.1 and Corollary 8.7. □

For Frege systems we have a similar result. Recall that a *Frege system* consists of some finite (but complete) set of axioms and the rule *modus ponens* that allows to deduce B from A and $\bar{A} \vee B$. We speak of an *extended Frege system* if we add extension axioms (17). Finally, we speak of a *Frege system with substitution* if we add the sub-rule (16) to a Frege system.

Theorem 9.6. *Substitution-free combinatorial flows are p-equivalent to Frege systems. Combinatorial flows (with substitution) are p-equivalent to Frege systems with substitution and to extended Frege systems.*

Proof. It has been shown in [BG09] (and [Str12]) that Frege systems (resp. Frege system with substitution, resp. extended Frege systems) are p-equivalent to SKS (resp. sSKS, resp. eSKS). Hence, the theorem follows from the results of the previous section. □

Remark 9.7. As before, it is straightforward to translate Frege proofs directly into combinatorial flows, without the detour via SKS: we first observe that every axiom can be translated into a simple combinatorial flow, and then the modus ponens rule can be simulated in the same way as the cut-rule in the sequent calculus. For the substitution rule we use the same construction as for sSKS in the proof of Theorem 8.8.

10. Conclusion and Future Work

In this paper we proposed a solution to the problem of finding syntax-independent presentations of classical proofs that can also cover proof compression mechanisms that are usually studied in the area of proof complexity. This way, they can serve as a notion of *proof certificate* [Mil11] that goes beyond mere cut-free sequent proofs.

Furthermore, the cut elimination presented in Section 6 can, together with the results of Section 8 also be used as an alternative normalization procedure for SKS derivations, since the normal forms are *streamlined* in the sense of [GG08] and [GGS10].

The obvious next step is to include first-order quantifiers in the presentation. There is already preliminary work by Hughes [Hug14] in this direction, but it still has to be investigated how the various notions of composition and normalization discussed in this paper behave in the presence of quantifiers.

Another direction of possible future research is question whether combinatorial flows can form some free category (in the same sense as MLL proof nets form the free unit-free star-autonomous category [HS16]) and the relation to categorical combinators [Cur86].

References

[And76] Peter B. Andrews. Refutations by matings. *IEEE Transactions on Computers*, C-25:801–807, 1976.

[BG09] Paola Bruscoli and Alessio Guglielmi. On the proof complexity of deep inference. *ACM Transactions on Computational Logic*, 10(2):1–34, 2009. Article 14.

[Bib81] Wolfgang Bibel. On matrices with connections. *Journal of the ACM*, 28:633–645, 1981.

[Boo84] George Boolos. Don't eliminate cut. *Journal of Philosophical Logic*, 13:373–378, 1984.

[BT01] Kai Brünnler and Alwen Fernanto Tiu. A local system for classical logic. In R. Nieuwenhuis and A. Voronkov, editors, *LPAR 2001*, volume 2250 of *LNAI*, pages 347–361. Springer, 2001.

[Bus91] Samuel R. Buss. The undecidability of k-provability. *Annals of Pure and Applied Logic*, 53:72–102, 1991.

[BvdW95] Gianluigi Bellin and Jacques van de Wiele. Subnets of proof-nets in MLL^-. In J.-Y. Girard, Y. Lafont, and L. Regnier, editors, *Advances in Linear Logic*, volume 222 of *London Mathematical Society Lecture Notes*, pages 249–270. Cambridge University Press, 1995.

[Car97] Alessandra Carbone. Interpolants, cut elimination and flow graphs for the propositional calculus. *Annals of Pure and Applied Logic*, 83:249–299, 1997.

[CPS85] Derek G. Corneil, Yehoshua Perl, and Lorna K. Stewart. A linear recognition algorithm for cographs. *SIAM J. Comput.*, 14(4):926–934, 1985.

[CR79] Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *The Journal of Symbolic Logic*, 44(1):36–50, 1979.

[Cur86] Pierre-Louis Curien. Categorical combinators. *Information and Control*, 69(1-3):188–254, 1986.

[Das13] Anupam Das. Rewriting with linear inferences in propositional logic. In Femke van Raamsdonk, editor, *24th International Conference on Rewriting Techniques and Applications (RTA)*, volume 21 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 158–173. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2013.

[DR89] Vincent Danos and Laurent Regnier. The structure of multiplicatives. *Annals of Mathematical Logic*, 28:181–203, 1989.

[Duf65] R.J Duffin. Topology of series-parallel networks. *Journal of Mathematical Analysis and Applications*, 10(2):303 – 318, 1965.

[Gen34] Gerhard Gentzen. Untersuchungen über das logische Schließen. I. *Mathematische Zeitschrift*, 39:176–210, 1934.

[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.

[GGS10] Alessio Guglielmi, Tom Gundersen, and Lutz Straßburger. Breaking paths in atomic flows for classical logic. In *LICS 2010*, 2010.

[Gir87] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[GS01] Alessio Guglielmi and Lutz Straßburger. Non-commutativity and MELL in the calculus of structures. In Laurent Fribourg, editor, *Computer Science Logic, CSL 2001*, volume 2142 of *LNCS*, pages 54–68. Springer-Verlag, 2001.

[Gue99] Stefano Guerrini. Correctness of multiplicative proof nets is linear. In *LICS*, pages 454–463, 1999.

[Gug07] Alessio Guglielmi. A system of interaction and structure. *ACM Transactions on Computational Logic*, 8(1):1–64, 2007.

[HS16] Willem Heijltjes and Lutz Straßburger. Proof nets and semi-star-autonomous categories. *Mathematical Structures in Computer Science*, 26(5):789–828, 2016.

[Hug05] Dominic Hughes. Simple multiplicative proof nets with units. Preprint, 2005.

[Hug06a] Dominic Hughes. Proofs Without Syntax. *Annals of Mathematics*, 164(3):1065–1076, 2006.

[Hug06b] Dominic Hughes. Towards Hilbert's 24^{th} problem: Combinatorial proof invariants: (preliminary version). *Electr. Notes Theor. Comput. Sci.*, 165:37–63, 2006.

[Hug14] Dominic Hughes. First-order proofs without syntax. Berkeley Logic Colloquium, 2014.

[KM71] Gregory Maxwell Kelly and Saunders Mac Lane. Coherence in closed categories. *J. of Pure and Applied Algebra*, 1:97–140, 1971.

[KP89] Jan Krajíček and Pavel Pudlák. Propositional proof systems, the consistency of first order theories and the complexity of computations. *The Journal of Symbolic Logic*, 54(3):1063–1079, 1989.

[Lau99] Olivier Laurent. Polarized proof-nets: proof-nets for LC (extended abstract). In Jean-Yves Girard, editor, *Typed Lambda Calculi and Applications (TLCA 1999)*, volume 1581 of *LNCS*, pages 213–227. Springer, 1999.

[Lau03] Olivier Laurent. Polarized proof-nets and $\lambda\mu$-calculus. *Theoretical Computer Science*, 290(1):161–188, 2003.

[LS05] François Lamarche and Lutz Straßburger. Naming proofs in classical propositional logic. In Paweł Urzyczyn, editor, *TLCA'05*, volume 3461 of *LNCS*, pages 246–261. Springer, 2005.

[Mil11] Dale Miller. A proposal for broad spectrum proof certificates. In J.-P. Jouannaud and Z. Shao, editors, *CPP: First International Conference on Certified Programs and Proofs*, volume 7086 of *Lecture Notes in Computer Science*, pages 54–69, 2011.

[Möh89] Rolf H. Möhring. Computationally tractable classes of ordered sets. In I. Rival, editor, *Algorithms and Order*, pages 105–194. Kluwer Acad. Publ., 1989.

[NS15] Novak Novakovic and Lutz Straßburger. On the power of substitution in the calculus of structures. *ACM Trans. Comput. Log.*, 16(3):19, 2015.

[Ret93] Christian Retoré. *Réseaux et Séquents Ordonnés*. PhD thesis, Université Paris VII, 1993.

[Ret03] Christian Retoré. Handsome proof-nets: perfect matchings and cographs. *Theoretical Computer Science*, 294(3):473–488, 2003.

[Rob03] Edmund P. Robinson. Proof nets for classical logic. *Journal of Logic and Computation*, 13:777–797, 2003.

[Str03a] Lutz Straßburger. *Linear Logic and Noncommutativity in the Calculus of Structures*. PhD thesis, Technische Universität Dresden, 2003.

[Str03b] Lutz Straßburger. MELL in the Calculus of Structures. *Theoretical Computer Science*, 309(1–3):213–285, 2003.

[Str07] Lutz Straßburger. A characterisation of medial as rewriting rule. In Franz Baader, editor, *Term Rewriting and Applications, RTA'07*, volume 4533 of *LNCS*, pages 344–358. Springer, 2007.

[Str11] Lutz Straßburger. From deep inference to proof nets via cut elimination. *Journal of Logic and Computation*, 21(4):589–624, 2011.

[Str12] Lutz Straßburger. Extension without cut. *Annals of Pure and Applied Logic*, 163(12):1995–2007, 2012.

[TS00] Anne Sjerp Troelstra and Helmut Schwichtenberg. *Basic Proof Theory*. Cambridge University Press, second edition, 2000.

[Tse68] G. S. Tseitin. On the complexity of derivation in propositional calculus. *Zapiski Nauchnykh Seminarou LOMI*, 8:234–259, 1968.

RESEARCH CENTRE
SACLAY – ÎLE-DE-FRANCE

1 rue Honoré d'Estienne d'Orves
Bâtiment Alan Turing
Campus de l'École Polytechnique
91120 Palaiseau

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399