HAL
archives-ouvertes.fr

Combinatorial Flows and their Normalisation

Lutz Straßburger

▶ **To cite this version:**

Lutz Straßburger. Combinatorial Flows and their Normalisation. FSCD 2017 - 2nd International Conference on Formal Structures for Computation and Deduction, Sep 2017, Oxford, United Kingdom. 84, pp.311 - 3117, 2017, Leibniz International Proceedings in Informatics (LIPIcs). <10.4230/LIPIcs.FSCD.2017.31>. <hal-01635931>

HAL Id: hal-01635931
https://hal.inria.fr/hal-01635931

Submitted on 15 Nov 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Combinatorial Flows and their Normalisation

Lutz Straßburger

Inria

Abstract

This paper introduces combinatorial flows that generalize combinatorial proofs such that they also include cut and substitution as methods of proof compression. We show a normalization procedure for combinatorial flows, and how syntactic proofs are translated into combinatorial flows and vice versa.

1998 ACM Subject Classification F.4.1 Mathematical Logic – Proof theory

Keywords and phrases proof equivalence, cut elimination, substitution, deep inference

Digital Object Identifier 10.4230/LIPIcs.FSCD.2017.31

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 formalisms*. This means that the first step must be to find ways to describe proofs independent of the formalisms, i.e., we need "canonical representations" which do not rely on some particular syntax of a chosen deductive formalism. For this reason, we also speak of "syntax-free" presentation of proofs.

The earliest attempts for such "syntax-free" proof presentations were Andrews' *matings* [1] and Bibel's *matrix proofs* [3] 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.

Girard's *proof-nets* for linear logic [11] were the first syntax-free proof presentation able to address these two issues. Proof nets can be seen as 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 [22, 28]. However, these proof-nets depend on a specific form of Gentzen's sequent calculus. They are neither able to capture proofs written in other sequent calculi, like G3c [32], nor other formalisms, like analytic tableaux or resolution.

This problem was addressed by *B-nets* [21], which exhibit a confluent cut elimination procedure and can capture proofs in most standard proof formalisms. However, their correctness


© Lutz Straßburger;
licensed under Creative Commons License CC-BY

criterion is exponential and the cut elimination cannot be lifted to the sequent calculus.

This issue has been addressed by *atomic flows* [12, 13] 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 [9] shows that there cannot be a polynomial correctness criterion for atomic flows, if integer factoring is hard for *P/poly*. The *C-nets* of [29], which are similar to atomic flows, but additionally form a closed category, have same problem.

Only the *combinatorial proofs* by Hughes [16] have a polynomial correctness criterion and are independent of any syntactic formalism. Cuts in combinatorial proofs are represented by formulas of the shape $A \wedge \bar{A}$ in the conclusion [17]. The cut elimination in [17] is then based on a form of projection combined with atomic substitution, and this construction largely inspired our vertical composition in Section 6.

Anyway, none of the existing "syntax-free" proof presentations can deal with proofs using *extension* or *substitution* [6, 20], which are, like the *cut*, methods of proof compression. They are mainly studied in the area of proof complexity, and only recently have received attention from structural proof theory [5, 30, 25].

The main contribution of this paper is a notion of "syntax-free" proof presentation that (1) comes with a polynomial correctness criterion, (2) is independent of the syntax of proof formalisms (like sequent calculi, tableaux systems, resolution, Frege systems, or deep inference systems), and (3) 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. Point (1) above is stated in Theorem 22, Point (3) is carried out in Sections 5 and 6. For Point (2), we sketch in Section 8 how deep inference proofs are translated into combinatorial flows. The technical report [31] also sketches how sequent proofs and Frege proofs can be translated into combinatorial flows.[1] In a future work we will show how this can be done for other fomalisms, like analytic tableaux or resolution. The proposed notion of proof identity here is that *"two proofs are the same if they have the same combinatorial flow"*.

2 Preliminaries on combinatorial proofs

Combinatorial proofs have been introduced by Hughes in [16] as a way to present proofs of classical logic independent of 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:

$$A, B ::= a \mid \bar{a} \mid A \wedge B \mid A \vee B \tag{1}$$

where $\bar{a}$ is the negation of a. The negation can then be defined for all formulas via $\bar{\bar{a}} = a$ and the De Morgan laws $\overline{A \vee B} = \bar{A} \wedge \bar{B}$ and $\overline{A \wedge B} = \bar{A} \vee \bar{B}$. Then it follows that $\bar{\bar{A}} = A$ for all formulas A. 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:

$$\Gamma = A_1, A_2, \ldots, A_n \tag{2}$$

[1] The report [31] also contains more technical details and missing proofs.

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 (resp. disjunction) of the formulas in Γ.

▸ Remark. 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.

Before we can discus the notion of combinatorial proof, we need some preliminary definitions.

▸ **Definition 1.** 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 in L, called its *label*. For two disjoint 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-labeled 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 $V_{\mathfrak{G}_1} \cap V_{\mathfrak{G}_2} = \varnothing$, 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. 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 $\mathfrak{G}(\Gamma) = \mathfrak{G}(\bigvee\Gamma) = \mathfrak{G}(A_1) \vee \mathfrak{G}(A_2) \vee \cdots \vee \mathfrak{G}(A_n)$.

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

▸ **Lemma 3.** *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 $\vee$:*

$$\begin{array}{ll} A \wedge (B \wedge C) = (A \wedge B) \wedge C & A \wedge B = B \wedge A \\ A \vee (B \vee C) = (A \vee B) \vee C & A \vee B = B \vee A \end{array} \tag{3}$$

Proof. Immediately from Construction 2. ◂

▸ **Example 4.** 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 $\mathfrak{G}(\bar{A}) = \overline{\mathfrak{G}(A)}$:

(4)

The following is well-known. It can already be found in [10] (see also [24, 26]).

▸ **Proposition 5.** *A graph $\mathfrak{G}$ is a cograph iff it can be constructed from a formula via Construction 2.*

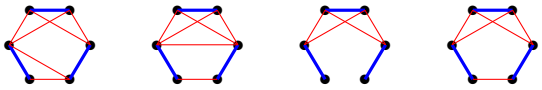
An important consequence of this and Lemma 3 is that for each cograph $\mathfrak{G}$ there is a unique (up to associativity and commutativity) formula tree determining $\mathfrak{G}$. We denote this formula tree by $F(\mathfrak{G})$.

▸ **Definition 6.** 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 7.** Let $\mathfrak{G} = \langle V_\mathfrak{G}, E_\mathfrak{G}\rangle$ be a multigraph. A set $B_\mathfrak{G} \subseteq E_\mathfrak{G}$ of edges is called a *matching* if no two edges in $B_\mathfrak{G}$ are adjacent. A matching $B_\mathfrak{G}$ is *perfect* if every vertex $v \in V_\mathfrak{G}$ is incident to an edge in $B_\mathfrak{G}$. An *R&B-graph* $\mathfrak{G} = \langle V_\mathfrak{G}, R_\mathfrak{G}, B_\mathfrak{G}\rangle$ is a triple such that $\langle V_\mathfrak{G}, R_\mathfrak{G} \uplus B_\mathfrak{G}\rangle$ is a multigraph such that $B_\mathfrak{G}$ is a perfect matching and $\langle V_\mathfrak{G}, R_\mathfrak{G}\rangle$ is a simple graph (i.e., $R_\mathfrak{G}$ is not allowed to have multiple edges). We will use the notation $\mathfrak{G}^\downarrow$ for the simple graph $\langle V_\mathfrak{G}, R_\mathfrak{G}\rangle$. An *R&B-cograph* is an R&B-graph $\mathfrak{G} = \langle V_\mathfrak{G}, R_\mathfrak{G}, B_\mathfrak{G}\rangle$ where $\mathfrak{G}^\downarrow = \langle V_\mathfrak{G}, R_\mathfrak{G}\rangle$ is a cograph.

As before, we omit the index $\mathfrak{G}$ when it is clear from context. Following [27] we will draw B-edges in blue/bold, and R-edges in red/regular. Below are four examples:



(5)

Also the next two definitions are taken from [27].

▸ **Definition 8.** A path (resp. cycle) in a multigraph 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 9.** 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 (5), 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 10.** 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 $wv \in B$ implies that the labels of $f(w)$ and $f(v)$ are dual to each other.

▸ **Definition 11.** 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 $f(w)w' \notin E'_\mathfrak{G}$.

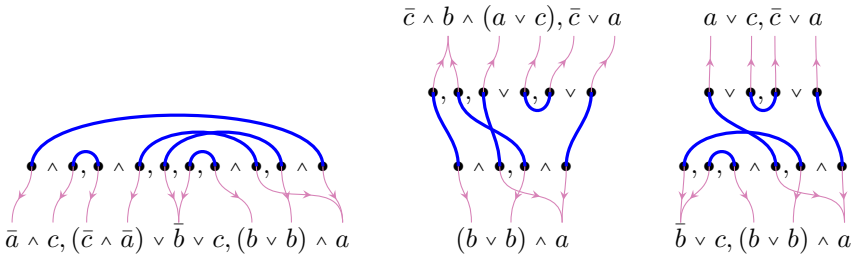


Figure 1 Examples of simple combinatorial flows (the cographs are obtained via Construction 2)

We are now ready to give the definition of a combinatorial proof together with the main result of [16].

▸ **Definition 12.** 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 $f\colon \mathfrak{C}^{\downarrow} \rightarrowtail \mathfrak{G}(\Gamma)$.

▸ Remark. Our presentation of the condition on the cograph in a combinatorial proof deters from Hughes' [16] and follows Retoré's [27] instead. The reason is that Retoré makes the relation to proof nets of linear logic [8] explicit. Also note, that the condition on the cograph $\mathfrak{C}^{\downarrow}$ given by Hughes [16, 17] 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 [27]. In our presentation here we also add the connectedness via chordless æ-paths in order to reject mix.

The two main results of [16] are that combinatorial proofs are sound and complete with respect to classical logic, and that they form a proof system in the sense of Cook and Reckhow [6].

▸ **Theorem 13** ([16]). *A formula is a theorem of classical propositional logic iff it has a combinatorial proof.*

▸ **Theorem 14** ([16]). *Given a 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 in the size of the input that $\mathfrak{C}$ and f form a combinatorial proof of A.*

3 Combinatorial flows

▸ **Definition 15.** 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.[2] 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 $|\mathfrak{C}^{\downarrow}| + |\Gamma| + |\Delta|$.

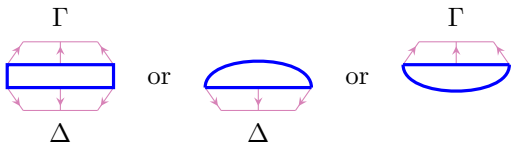
▸ **Lemma 16.** *Let $\mathfrak{C}$, $\mathfrak{G}_1$, and $\mathfrak{G}_2$ be cographs and let $f\colon \mathfrak{C} \rightarrowtail \mathfrak{G}_1 \vee \mathfrak{G}_2$ be a skew fibration. Then there are cographs $\mathfrak{C}_1$ and $\mathfrak{C}_2$ and graph homomorphisms $f_1\colon \mathfrak{C}_1 \rightarrowtail \mathfrak{G}_1$ and $f_2\colon \mathfrak{C}_2 \rightarrowtail \mathfrak{G}_2$ such that $\mathfrak{C} = \mathfrak{C}_1 \vee \mathfrak{C}_2$ and $f = f_1 \vee f_2$.*

▸ Notation 17. This lemma allows us to depict simple 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

[2] Note that it cannot happen that both Γ and Δ are empty.

$\mathfrak{C}_\Gamma$ and $\mathfrak{C}_\Delta$ be the cographs determined by Lemma 16 (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 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.

Schematically we can depict simple combinatorial flows as follows:

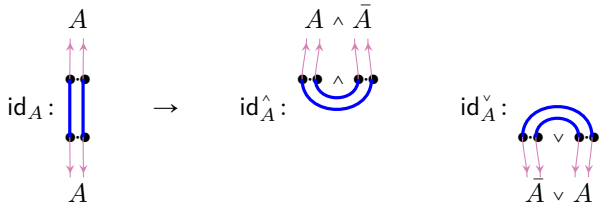


where the middle and the right picture are used to indicate that Γ or Δ, respectively, are empty.

▸ **Lemma 18.** *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 15. ◂

▸ Observation 19. 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 18 to id_A we get two simple combinatorial flows $\mathsf{id}_A^{\wedge} \colon A \wedge \bar{A} \vdash \circ$ and $\mathsf{id}_A^{\vee} \colon \circ \vdash \bar{A} \vee A$, as depicted below:



(6)

▸ **Definition 20.** 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 $\bar{B}_i$. Then, for substitution proof into proofs, we also need a notation for formula substitutions in which a 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 21.** The set of *combinatorial flows* is defined inductively as follows:

- A simple combinatorial flow $\phi \colon A \vdash B$ is a combinatorial flow.

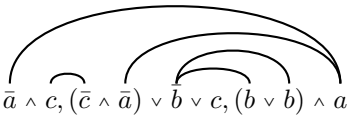
- If $\phi\colon A \vdash B$ and $\psi\colon C \vdash D$ are combinatorial flows then so are $\phi \wedge \psi\colon A \wedge B \vdash C \wedge D$ and $\phi \vee \psi\colon A \vee C \vdash B \vee D$. This operation is called *horizontal composition*.
- If $\phi\colon \Gamma \vdash A$ and $\psi\colon A \vdash \Delta$ are combinatorial flows then $\phi \blacklozenge \psi\colon \Gamma \vdash \Delta$ 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 22.** *Combinatorial flows form a proof system (in the sense of [6]). In particular, checking correctness of a combinatorial flow can be done is polynomial time.*

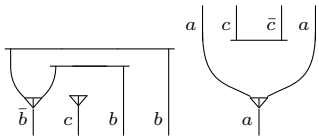
Proof. This follows immediately from Theorem 14, Definition 15, and Definition 21. ◂

▸ Remark. Theorem22 provides the main advantage of combinatorial flows over B-nets and N-nets [21] and atomic flows [12, 13]. 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:



(7)

The corresponding B-net is obtained by forgetting the multiplicity of the edges. In the example in (7), 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.

▸ **Definition 23.** 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 21. The following four sections are dedicated to this.

4 Normalization I: Binary Connectives

▸ **Lemma 24.** *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 $|\xi| \leqslant |\phi| + |\psi|$.*

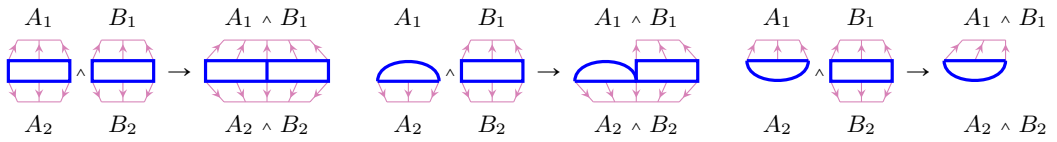


Figure 2 Conjunction of simple combinatorial flows

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 16. 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 $B_{\mathfrak{H}} = B_{\mathfrak{C}} \cup B_{\mathfrak{D}}$, 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 $h = f$.
- 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 $h = g$.

Then, $\mathfrak{H}$ is an R&B-cograph (by construction) and it is critically chorded. In the first case the situation is the same as in the $\otimes$-rule for MLL-proof nets (see [27]) 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. 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. Note also that there is an arbitrary choice to make when both $\mathfrak{C}_2$ and $\mathfrak{D}_2$ are empty.

5 Normalization II: Substitution

▸ **Lemma 25.** *Let $\phi\colon \Gamma \vdash \Delta$ and $\psi\colon C \vdash D$ be simple combinatorial flows. Then there is a simple combinatorial flow $\phi'\colon \Gamma[a/C, \bar{a}/\bar{D}] \vdash \Delta[a/D, \bar{a}/\bar{C}]$.*

This is depicted on the left of Figure 3. The basic idea of 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 26. 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 27.** *If $\mathfrak{C}$ and $\mathfrak{D}$ are cographs and $x \in V_{\mathfrak{C}}$, then $\mathfrak{C}[x/\mathfrak{D}]$ is also a cograph.*

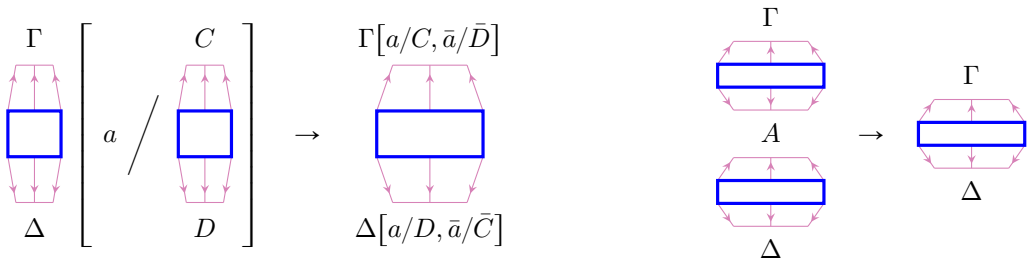


Figure 3 Left: Substitution elimination Right: cut elimination

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 5. ◂

▸ Construction 28. In Construction 26 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 26 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 and replaced by the matching $B_{\mathfrak{D}}$.

▸ **Lemma 29.** *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 27. 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 25. Let ϕ and ψ as above and let $\Gamma' = \Gamma[a/C, \bar{a}/\bar{D}]$ and $\Delta' = \Delta[a/D, \bar{a}/\bar{C}]$. 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 16, 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 28 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

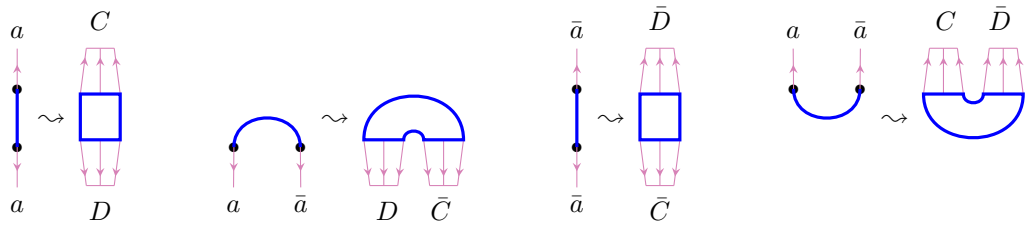


Figure 4 Substitution of simple combinatorial flows

$f'(z) = f(z)$. 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, as indicated on the right of Figure 3. This is done via "projection + atomic substitution", as introduced in [17], but refined in such a way that we do not need mix and the notion of "laxness".

▸ **Lemma 30.** *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, we need first to establish some preliminary properties on skew fibrations and the composition of R&B-cographs.

▸ **Lemma 31.** *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 of $\mathfrak{C}$, $\mathfrak{D}$, $\mathfrak{G}$, and $\mathfrak{H}$ 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 32. 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 $\mathfrak{E} = \mathfrak{C} \blacklozenge \mathfrak{D}$ 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 33.** *If in Construction 32 the R&B-cographs $\mathfrak{C}$ and $\mathfrak{D}$ are critically chorded, then so is $\mathfrak{E} = \mathfrak{C} \blacklozenge \mathfrak{D}$.*

Proof. This follows directly from the correspondence to MLL$^-$ proof nets given in [27] and the standard cut elimination result for linear logic proof nets. The idea used here goes back to [19], and a more recent presentation can be found in [15]. ◂

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.

▸ Construction 34. Let $\phi\colon \Gamma \vdash B \wedge C$ be given by a critically chorded 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

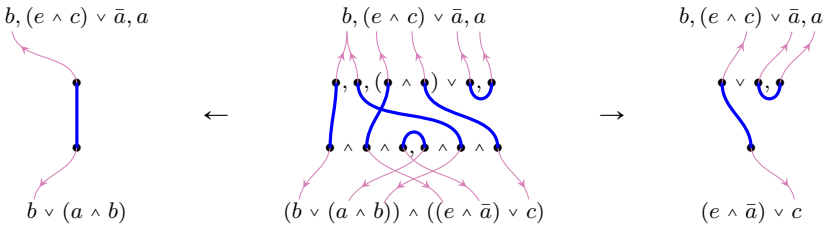
- If $x \in U_C$ and $xy \in B_{\mathfrak{C}}$ and $y \notin U_C$ then $y \in U_C^{\perp}$.
- If $x \in U_C^{\perp}$ and $xy \in B_{\mathfrak{C}}$ and $y \notin U_C$ then $y \in U_C^{\perp}$.
- If $V', V'' \subseteq V_{\mathfrak{C}}$ induce subcographs and $V' \subseteq U_C^{\perp}$ and $V' \cap V'' = \varnothing$ 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 $V_{\mathfrak{C}_l}$.[4]

The idea behind this construction is to remove from $\mathfrak{C}$ the preimage of C together with the largest "critically chorded sub-(R&B-cograph)" that contains all vertices to which there is a B-edge from any vertex in the preimage of C.

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 $V_{\mathfrak{C}_l} \neq \varnothing$. To see that, let $U_B \subseteq V_{\mathfrak{C}}$ be the set of all vertices in $\mathfrak{C}$ that are mapped by f to atom occurrences in B. Now note that either both U_B and U_C are empty or both are not empty (because f is skew). If both U_B and U_C are empty, then $V_{\mathfrak{C}_l} = V$ which is not empty by definition. Otherwise, if U_B and U_C are both nonempty, but $V_{\mathfrak{C}_l}$ is, then all of U_B must be contained in $U_C^{\perp}$. Furthermore, at least one æ-paths connecting U_B and U_C starts and ends with a B-edge. Hence, an æ-cycle is closed by the R-edge between the two end vertices (because of the $\wedge$-connective between B and C), contradicting that $\mathfrak{C}$ is critically chorded.

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 $\phi_r\colon \Gamma \vdash C$. Below is an example of a simple flow and its two projections:



(8)

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 $\psi_r\colon C \vdash \Delta$.

[3] This step can be seen as a combination of the ↓-, ⇓-, and ↑-steps in the empire construction in [2].
[4] As mentioned before, this construction is different from the one in [17], due to the absence of mix and "laxness". However, it remains open, how this compares to the "greedy garbage collection" of [17].

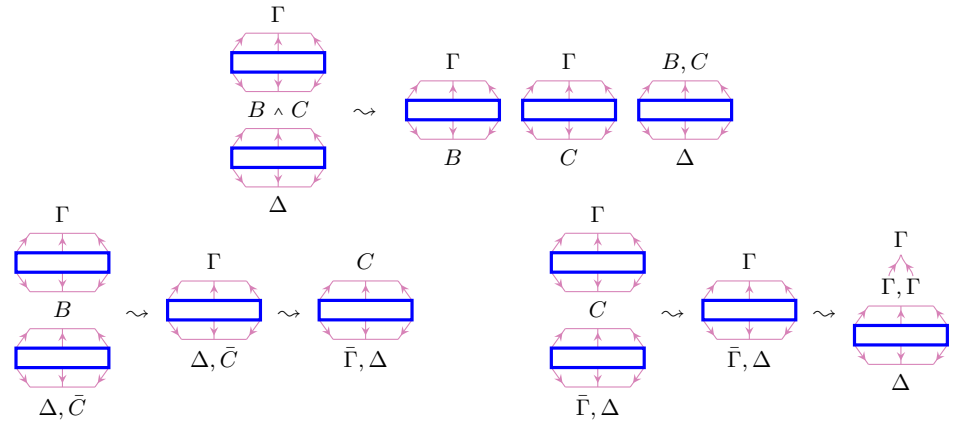
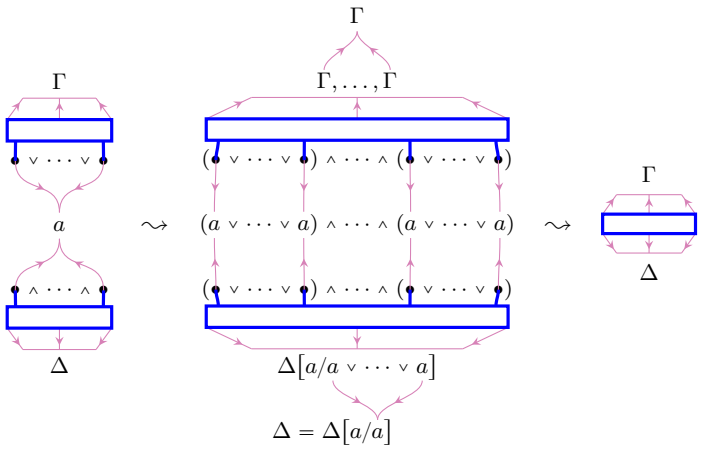


Figure 5 Steps in the proof of Lemma 30

Proof of Lemma 30. 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 $\psi'\colon B, C \vdash \Delta$ (see top line of Figure 5). From ψ' we can obtain (via Lemma 18) $\psi''\colon B \vdash \Delta, \bar{C}$, 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 18) we can get a simple flow $\chi'\colon C \vdash \bar{\Gamma}, \Delta$, as shown on the lower left of Figure 5. 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 18. Finally, we can apply Lemma 31 to get the desired $\chi\colon \Gamma \vdash \Delta$, as shown on the lower right Figure 5. If $A = B \vee C$ we proceed analogous. It remains to show the case when A is an atom, for which the construction is depicted below:



(9)

Here, let $f\colon \mathfrak{C}^{\downarrow} \rightarrowtail \mathfrak{G}(\wedge\bar{\Gamma}, a)$ and $g\colon \mathfrak{D}^{\downarrow} \rightarrowtail \mathfrak{G}(\bar{a}, \vee\Delta)$ 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 $f^*\colon \mathfrak{C}^{\downarrow} \rightarrowtail \mathfrak{G}(\wedge\bar{\Gamma}, a \vee \cdots \vee a)$ where we replace a by a disjunction of n copies of a, and let f^* behave as f on $V_{\mathfrak{C}}\backslash\{x_1, \ldots, x_n\}$ 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 $g^*\colon \mathfrak{D}^{\downarrow} \rightarrowtail \mathfrak{G}(\bar{a} \vee \cdots \vee \bar{a}, \vee\Delta)$ 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 then apply the construction of Section 4 to form the conjunction of m copies of ϕ^*, which yields a simple flow $\hat{\phi}\colon \Gamma, \dots, \Gamma \vdash (a \vee \cdots \vee a) \wedge \cdots \wedge (a \vee \cdots \vee a)$.

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 $\hat{\psi}\colon (a \vee \cdots \vee a) \wedge \cdots \wedge (a \vee \cdots \vee a) \vdash \Delta[a/a \vee \cdots \vee a]$.[5]

Finally, we plug $\hat{\phi}$ and $\hat{\psi}$ together and apply Lemma 33 to get a simple flow $\chi'\colon \Gamma, \dots, \Gamma \vdash \Delta[a/a \vee \cdots \vee a]$, to which we apply Lemma 31 to get the desired simple flow $\chi\colon \Gamma \vdash \Delta$. ◂

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 24, 25, and 30, then we have immediately the following:

▸ **Theorem 35.** *The relation $\to$ is strongly normalizing, and the normal forms are 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 36.** *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.*

8 Relation to to deep inference proofs

In this section we show how combinatorial flows are related to syntactic proofs in deep inference. It should be clear that the similar constructions are possible with other proof formalisms (like tableaux, sequent calculus, or resolution) as well.

We use the version of the deep inference system SKS [4], which is shown in Figure 6.[6] The rules shown there should be read as rewrite rule schemes that can be applied inside an arbitrary (positive) formula context. In the rules $\mathsf{ai}{\downarrow}$, $\mathsf{ai}{\uparrow}$, $\mathsf{ac}{\downarrow}$, and $\mathsf{ac}{\uparrow}$, the a can stand for any atom. In all rules, A, B, C, and D, can stand for any formula, and in $\mathsf{ai}{\downarrow}$ we additionally allow A to be empty, so to have proper proofs without premise.[7] We write

$$\begin{array}{c} P \\ \mathsf{s}\big\|_{\Phi} \\ Q \end{array} \qquad \text{and} \qquad \begin{array}{c} \mathsf{s}\big\|_{\Psi} \\ Q \end{array} \tag{10}$$

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 (3). Figure 9 shows on the left an example of a derivation in SKS, where $2 \cdot \mathsf{s}$ stands for two consecutive applications of the s-rule.

Each rule in system SKS can straightforwardly be translated into a simple combinatorial flow, as indicated in Figure 7, where the double lines indicate the identity (see Observation 19).

---

[5] 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).

[6] Note that our system is slightly different from the original version of SKS in [4]: We do not have explicit units in the language, and therefore our weakening rule is not atomic (see also [30]).

[7] We could also allow A to be empty in $\mathsf{ai}{\uparrow}$, so to have a proper refutation without conclusion.

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

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

Figure 6 Deep inference system SKS

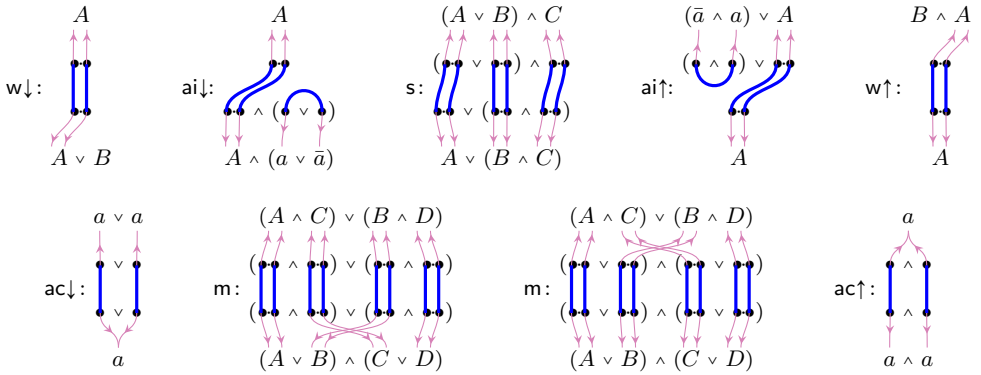


Figure 7 Simple combinatorial flows for the rules in Figure 6

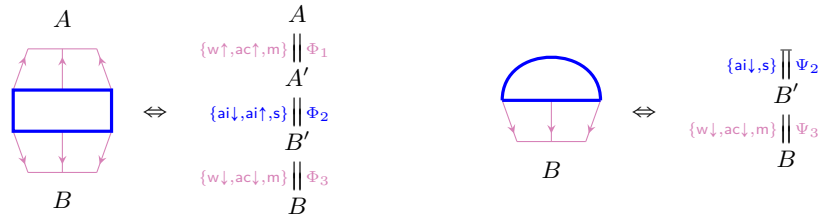


Figure 8 Relation between simple combinatorial flows and SKS derivations

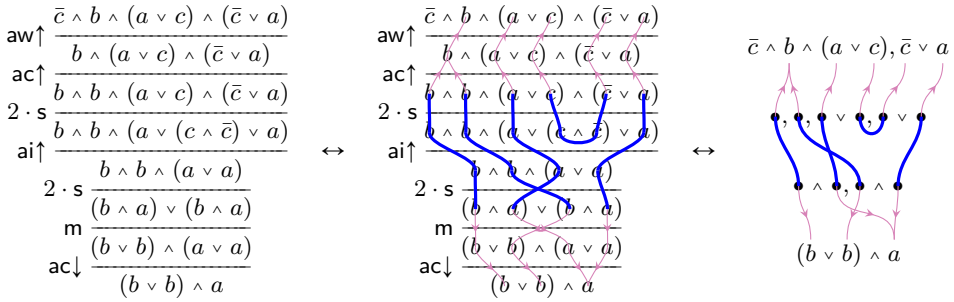


Figure 9 Example of an SKS derivation and its simple combinatorial flow

Note that for the m-rule there are two possible translations. Since whenever $A = B$ modulo associativity and commutativity (3) 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 first direction of the following:

▸ **Theorem 37.** *Substitution-free combinatorial flows and system* SKS *p-simulate each other.*

For the other direction we use the relation between simple combinatorial flows and system SKS shown in Figure 8 to translate simple flows into SKS derivations which can be composed horizontally and vertically. Figure 9 shows the corresponding SKS derivations for the second example in Figure 1. For a detailed proof see [31].

Let us now investigate what happens when substitution is present. The substitution rule in a deductive system is given as follows:

$$\mathsf{sub}\,\frac{A}{A\sigma} \tag{11}$$

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 6) the rule sub in (11) 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. It has recently been established that sSKS is p-equivalent to Frege systems with substitution and Frege systems with extension [5, 30, 25]. Here we establish that combinatorial flows have the same expressiveness with respect to p-simulation:

▸ **Theorem 38.** *Combinatorial flows and* sSKS *p-simulate each other.*

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.

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 [33], following the ideas used in [30]. For a detailed proof see [31].

9 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* [23] 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 [12] and [13].

The obvious next step is to include first-order quantifiers in the presentation. There is already preliminary work by Hughes [18] 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 the 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 [14]) and the relation to categorical combinators [7].

Acknowledgements

I thank Anupam Das and Alessio Guglielmi for fruitful discussions, and I thank Paola Bruscoli, Dominic Hughes, and anonymous referees for helpful comments on earlier drafts of this work.

References

1 Peter B. Andrews. Refutations by matings. *IEEE Transactions on Computers*, C-25:801–807, 1976.
2 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.
3 Wolfgang Bibel. On matrices with connections. *Journal of the ACM*, 28:633–645, 1981.
4 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.
5 Paola Bruscoli and Alessio Guglielmi. On the proof complexity of deep inference. *ACM Transactions on Computational Logic*, 10(2):1–34, 2009. Article 14.
6 Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *The Journal of Symbolic Logic*, 44(1):36–50, 1979.
7 Pierre-Louis Curien. Categorical combinators. *Information and Control*, 69(1-3):188–254, 1986. doi:10.1016/S0019-9958(86)80047-X.
8 Vincent Danos and Laurent Regnier. The structure of multiplicatives. *Annals of Mathematical Logic*, 28:181–203, 1989.
9 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.
10 R.J Duffin. Topology of series-parallel networks. *Journal of Mathematical Analysis and Applications*, 10(2):303 – 318, 1965.
11 Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.
12 Alessio Guglielmi and Tom Gundersen. Normalisation control in deep inference via atomic flows. *Logical Methods in Computer Science*, 4(1:9):1–36, 2008. URL: http://arxiv.org/abs/0709.1205.
13 Alessio Guglielmi, Tom Gundersen, and Lutz Straßburger. Breaking paths in atomic flows for classical logic. In *LICS 2010*, 2010.
14 Willem Heijltjes and Lutz Straßburger. Proof nets and semi-star-autonomous categories. *Mathematical Structures in Computer Science*, 26(5):789–828, 2016. doi:10.1017/S0960129514000395.
15 Dominic Hughes. Simple multiplicative proof nets with units. Preprint, 2005. URL: http://arxiv.org/abs/math.CT/0507003.
16 Dominic Hughes. Proofs Without Syntax. *Annals of Mathematics*, 164(3):1065–1076, 2006.
17 Dominic Hughes. Towards Hilbert's 24$^{\text{th}}$ problem: Combinatorial proof invariants: (preliminary version). *Electr. Notes Theor. Comput. Sci.*, 165:37–63, 2006.
18 Dominic Hughes. First-order proofs without syntax. Berkeley Logic Colloquium, 2014.
19 Gregory Maxwell Kelly and Saunders Mac Lane. Coherence in closed categories. *J. of Pure and Applied Algebra*, 1:97–140, 1971.
20 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.

21 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. URL: `http://www.lix.polytechnique.fr/~lutz/papers/namingproofsCL.pdf`.

22 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.

23 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.

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

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

26 Christian Retoré. *Réseaux et Séquents Ordonnés*. PhD thesis, Université Paris VII, 1993.

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

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

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

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

31 Lutz Straßburger. Combinatorial Flows and Proof Compression. Research Report RR-9048, Inria Saclay, 2017. URL: `https://hal.inria.fr/hal-01498468`.

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

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