Combinatorial Proofs and Decomposition Theorems for First-order Logic
Dominic J D Hughes, Lutz Strassburger, Jui-Hsuan Wu

To cite this version:
Dominic J D Hughes, Lutz Strassburger, Jui-Hsuan Wu. Combinatorial Proofs and Decomposition Theorems for First-order Logic. 2021 36th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), Jun 2021, Rome, Italy. pp.1-13, 10.1109/LICS52264.2021.9470579. hal-03369764

HAL Id: hal-03369764
https://inria.hal.science/hal-03369764
Submitted on 7 Oct 2021

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.

Distributed under a Creative Commons Attribution 4.0 International License
Combinatorial Proofs and Decomposition Theorems for First-order Logic

Dominic J. D. Hughes
Logic Group
U.C. Berkeley
USA

Lutz Straßburger
Inria, Equipe Partout
Ecole Polytechnique, LIX
France

Jui-Hsuan Wu
Ecole Normale Supérieure
France

Abstract—We uncover a close relationship between combinatorial and syntactic proofs for first-order logic (without equality). Whereas syntactic proofs are formalized in a deductive proof system based on inference rules, a combinatorial proof is a syntax-free presentation of a proof that is independent from any set of inference rules. We show that the two proof representations are related via a deep inference decomposition theorem that establishes a new kind of normal form for syntactic proofs. This yields (a) a simple proof of soundness and completeness for first-order combinatorial proofs, and (b) a full completeness theorem: every combinatorial proof is the image of a syntactic proof.

I. INTRODUCTION

First-order predicate logic is a cornerstone of modern logic. Since its formalisation by Frege [1] it has seen a growing usage in many fields of mathematics and computer science. Upon the development of proof theory by Hilbert [2], proofs became first-class citizens as mathematical objects that could be studied on their own. Since Gentzen’s sequent calculus [3], [4], many other proof systems have been developed that allow the implementation of efficient proof search, for example analytic tableaux [5] or resolution [6]. Despite the immense progress made in proof theory in general and in the area of automated and interactive theorem provers in particular, we still have no satisfactory notion of proof identity for first-order logic. In this respect, proof theory is quite different from any other mathematical field. For example in group theory, two groups are the same iff they are isomorphic; in topology, two spaces are the same iff they are homeomorphic; etc. In proof theory, we have no such notion telling us when two proofs are the same, even though Hilbert was considering this problem as a possible 24th problem [7] for his famous lecture in 1900 [8], before proof theory existed as a mathematical field.

The main reason for this problem is that formal proofs, as they are usually studied in logic, are inextricably tied to the syntactic (inference rule based) proof system in which they are carried out. And it is difficult to compare two proofs that are produced within two different syntactic proof systems, based on different sets of inference rules. Consider the derivations in Figure 1 showing two proofs of the formula \((\forall x \lor (\forall y.p))\), in sequent calculus (top) and in a deep inference system (bottom). It is, a priori, not clear how to compare them.

[Long version of the LICS 2021 paper, with full proofs in the appendix.]
This is where combinatorial proofs come in. They were introduced by Hughes [9] for classical propositional logic as a syntax-free notion of proof, and as a potential solution to Hilbert’s 24th problem [10] (see also [11]). The basic idea is to abstract away from the syntax of the inference rules used in inductively-generated proofs and consider the proof as a combinatorial object, more precisely as a special kind of graph homomorphism. For example, a propositional combinatorial proof of Peirce’s law \((p \Rightarrow q) \Rightarrow p\) is shown mid-left in Fig. 1. A homomorphism from a 4-vertex graph with two colours (above) to a 4-vertex graph labelled with propositional variables (below); dotted vertical lines define the homomorphism, from above to below.

Several authors have illustrated how syntactic proofs in various proof systems can be translated to propositional combinatorial proofs: for sequent proofs in [10], for deep inference proofs in [12], for Frege systems in [13], and for tableaux systems and resolution in [14]. This enables a natural definition of proof identity for propositional logic: two proofs are the same if they are mapped to the same combinatorial proof. A prominent example of a decomposition theorem is Herbrand’s theorem [21], which allows a separation between the propositional part and the quantifier part in a first-order proof [4], [22]. Through the advent of deep inference, new kinds of proof decompositions became possible, most notably the separation between the linear part of a proof and the resource management of a proof. It has been shown by Straßburger [23] that a proof in classical propositional logic can be decomposed into a proof of multiplicative linear logic, followed by a proof consisting only of contractions and weakenings (see also [10] §4). In this paper we show that the same is possible for first-order logic.

Combinatorial proofs and deep inference can be seen as opposite ends of a spectrum: whereas deep inference allows for a very fine granularity of inference rules—one inference rule in a standard formalism, like sequent calculus or semantic tableaux, is usually simulated by a sequence of different deep inference rules—combinatorial proofs have completely abolished the concept of inference rule. And yet, there is a close relationship between the two, realized through a decomposition theorem, as we establish in this paper.

Outline: This paper has three parts. First, in Sections II-V we present the preliminaries on first-order logic, first-order graphs, first-order combinatorial proofs, and the first-order proof system KS1. Second, in Section VII we state the main results. And third, in Sections VIII-X we give their proofs.

II. Preliminaries: First-Order Logic

A. Terms and Formulas

Fix pairwise disjoint countably infinite sets \(\text{VAR} = \{x, y, z, \ldots\}\) of variables, \(\text{FUN} = \{f, g, \ldots\}\) of function symbols, and \(\text{PRED} = \{p, q, \ldots\}\) of predicate symbols. Each function symbol and each predicate symbol has a finite arity. Each predicate symbol \(p\) has a dual \(\overline{p}\) with \(\overline{\overline{p}} \neq p\). The grammars below generate the set \(\text{TERM}\) of terms, denoted by \(t, s, u, \ldots\), the set \(\text{ATOM}\) of atoms, denoted by \(a, b, c, \ldots\), and the set \(\text{FORMULAS}\) of formulas, denoted by \(A, B, C, \ldots\):

\[
\begin{align*}
t & ::= x \mid f(t_1, \ldots, t_n) \\
a & ::= t \mid f \mid p(t_1, \ldots, t_n) \mid \overline{p}(t_1, \ldots, t_n) \\
A & ::= a \mid A \land A \mid A \lor A \mid \exists x. A \mid \forall x. A
\end{align*}
\]

where the arity of \(f\) and \(p\) is \(n\). For better readability we often omit parentheses and write \(f_{t_1 \ldots t_n}\) or \(p_{t_1 \ldots t_n}\). We consider the truth constants \(t\) (true) and \(f\) (false) as additional atoms, and consider all formulas in negation normal form, where negation \(\overline{\cdot}\) is defined on atoms and formulas via De Morgan’s laws:

\[
\begin{align*}
\overline{t} & = f \\
\overline{f} & = t \\
\overline{\exists x. A} & = \forall x. \overline{A} \\
\overline{\forall x. A} & = \exists x. \overline{A} \\
\overline{A \land B} & = A \lor B \\
\overline{A \lor B} & = A \land B
\end{align*}
\]

Note \(\overline{\overline{A}} = A\). We write \(A \Rightarrow B\) as an abbreviation for \(\overline{A} \lor B\).
A formula is **rectified** if all bound variables are distinct from one another and from all free variables. Every formula can be transformed into a logically equivalent rectified form by bound variable renaming, e.g. \((\forall x \forall \exists x q x) \land \exists r r \mapsto (\forall x \forall \exists q y q y) \land \exists r z z\). If we consider formulas equivalent modulo bound variable renaming (\(\alpha\)-conversion), the rectified form of a formula \(A\) is unique, and we denote it by \(\hat{A}\).

A **substitution** is a function \(\sigma: \text{VAR} \to \text{TERM}\) that is the identity almost everywhere. We denote substitutions as \(\sigma = [x_1/t_1, \ldots, x_n/t_n]\), where \(\sigma(x_i) = t_i\) for \(i = 1..n\) and \(\sigma(x) = x\) for all \(x \notin \{x_1, \ldots, x_n\}\). Write \(\text{As}\) for the formula obtained from \(A\) by applying \(\sigma\), i.e., by simultaneously replacing all occurrences of \(x_i\) by \(t_i\). A **variable renaming** is a substitution \(\rho\) with \(\rho(x) \in \text{VAR}\) for all variables \(x\).

**B. Sequent Calculus LK1**

**Sequents**, denoted by \(\Gamma, \Delta, \ldots\), are finite multisets of formulas, written as lists, separated by comma. The **corresponding formula of** a (non-empty) sequent \(\Gamma = A_1, A_2, \ldots, A_n\) is the disjunction of its formulas: \(\bigvee(\Gamma) = A_1 \lor A_2 \lor \cdots \lor A_n\).

\(\Gamma\) is a **sequent** if it is provable in \(\text{LK1}\).

\[\Gamma \vdash \Delta\]

\(\text{The cut elimination theorem. The cut rule is}
\]
\[\text{cut} \quad \vdash \Gamma, A \quad \vdash \Delta, \quad \vdash \Gamma, \Delta \]
\[\Gamma \vdash \Delta\]

**Theorem 1.** LK1 is sound and complete for first-order logic.

For a proof, see any standard textbook, e.g. [24].

The linear fragment of LK1, i.e., the fragment without the rules \texttt{ctr} (contraction) and \texttt{wk} (weakening) defines first-order multiplicative linear logic [19], [25] with mix [26], [27] (MLL1+mix). We denote that system here with MLL1\(^X\) (shown in Figure 2 in the dashed box).

We will use the cut elimination theorem. The cut rule is

\[\vdash \Gamma, A \quad \vdash \Delta, \quad \vdash \Gamma, \Delta \]

\[\Gamma \vdash \Delta\]

**Theorem 2.** If a sequent \(\vdash \Gamma\) is provable in LK1+cut then it is also provable in LK1. Furthermore, if \(\vdash \Gamma\) is provable in MLL1\(^X\)+cut then it is also provable in MLL1\(^X\).

As before, this is standard, see e.g. [24] for a proof.

**III. PRELIMINARIES: FIRST-ORDER GRAPHS**

**A. Graphs**

A graph \(\Gamma = (V_G, E_G)\) is a pair where \(V_G\) is a finite set of vertices and \(E_G\) is a finite set of edges, which are two-element subsets of \(V_G\). We write \(vw\) for an edge \(\{v, w\}\).

Let \(\Gamma = (V_G, E_G)\) and \(\mathcal{H} = (V_H, E_H)\) be graphs such that \(V_G \cap V_H = \emptyset\). A **homomorphism** \(\varphi: \Gamma \to \mathcal{H}\) is a function \(\varphi: V_G \to V_H\) such that if \(vw \in E_G\) then \(\varphi(v)\varphi(w) \in E_H\).

The **union** \(\Gamma + \mathcal{H}\) is the graph \(\{V_G \cup V_H, E_G \cup E_H\}\) and the **join** \(\Gamma \times \mathcal{H}\) is the graph \(\{V_G \cup V_H, E_G \cup E_H \cup \{vw \mid v \in V_G, w \in V_H\}\}\). A graph \(\Gamma\) is **disconnected** if \(\Gamma = G_1 + G_2\) for two non-empty graphs \(G_1, G_2\); otherwise it is **connected**.

A graph \(\Gamma\) is **labelled** in a set \(L\) if each vertex \(v \in V_G\) has an associated **label** \(\ell(v) \in L\). A graph \(\Gamma\) is (partially) **coloured** if it carries a partial equivalence relation \(\sim\) on \(V_G\); each equivalence class is a **colour**. A **vertex renaming** is a function \(\rho: V_G \to V_G\) along a bijection \(\hat{\rho}\) that is the **vertex renaming**. A **directed graph** \(\Gamma = (V_G, E_G)\) along a bijection \((\hat{\gamma}): V_G \to V_H\) is the graph \(\hat{\gamma} = (V_G, \{\hat{\gamma}vw \mid vw \in E_G\})\), with colouring and/or labelling inherited (i.e., \(\hat{\gamma}v \sim \hat{\gamma}w\) if \(v \sim w\), and \(\hat{\gamma}\ell(v) = \ell(v)\)).

Following standard graph theory, we identify graphs modulo vertex renaming.

A **directed graph** \(\Gamma = (V_G, E_G)\) is a set \(V_G\) of vertices and a set \(E_G \subseteq V_G \times V_G\) of **direct edges**. A **directed graph homomorphism** \(\varphi: (V_G, E_G) \to (V_H, E_H)\) is a function \(\varphi: V_G \to V_H\) such that \((v, w) \in E_G\) then \((\varphi(v), \varphi(w)) \in E_H\).

**B. Cographs**

A graph \(\mathcal{H} = (V_H, E_H)\) is a **subgraph** of a graph \(\Gamma = (V_G, E_G)\) if \(V_H \subseteq V_G\) and \(E_H \subseteq E_G\). It is **induced** if \(v, w \in V_H\) and \(vw \in E_G\) implies \(vw \in E_H\). An induced subgraph of \(\Gamma = (V_G, E_G)\) is uniquely determined by its set of vertices \(V\) and we denote it by \(\Gamma[V]\). A graph is **\(\mathcal{H}\)-free** if it does not contain \(\mathcal{H}\) as an induced subgraph. The graph \(P_4\) is the (undirected) graph \(\{\{v_1, v_2, v_3, v_4\}, \{v_1v_2, v_2v_3, v_3v_4\}\}\). A **cograph** is a \(P_4\)-free undirected graph. The interest in cographs for our paper comes from the following well-known fact.

**Theorem 3** ([28], [29]). A graph is a cograph if it can be constructed from the singletons via the operations \(+\) and \(\times\).

In a graph \(\Gamma\), the **neighbourhood** \(N(v)\) of a vertex \(v \in V_G\) is \(\{w \mid vw \in E_G\}\). A **module** is a set \(M \subseteq V_G\) with \(N(v) \setminus M = N(w) \setminus M\) for all \(v, w \in M\). A module is **strong** if for every module \(M'\) we have \(M' \subseteq M\), \(M \subseteq M'\) or \(M \cap M' = \emptyset\). A module is **proper** if it has two or more vertices.

**C. Fographs**

A cograph is **logical** if every vertex is labelled by either an atom or variable, and it has at least one atom-labelled vertex. An atom-labelled vertex is a **literal** and a variable-labelled vertex is a **binder**. A binder labelled with \(x\) is an **\(x\)-binder**. The **scope** of a binder \(b\) is the smallest proper strong module 1

---

1In [10] and [13] adjacent vertices must have distinct colours, following the standard definition of colouring in graph theory. We choose to omit this condition here, as it is implied by the preclusion of bimatchings in Def. [10].
containing $b$. An $x$-literal is a literal whose atom contains the variable $x$. An $x$-binder binds every $x$-literal in its scope. In a logical cograph $G$, a binder $b$ is existential (resp. universal) if, for every other vertex $v$ in its scope, we have $bv \in E_G$ (resp. $bv \notin E_G$). An $x$-binder is legal if its scope contains no other $x$-binder and at least one literal.

**Definition 4 ([I8] §3).** A first-order graph or fograph $G$ is a logical cograph whose binders are all legal. The binding graph of $G$ is the directed graph $\vec{G} = \langle V_G, \{ (b, l) \mid b \text{ binds } l \} \rangle$.

We define a mapping $\llbracket \cdot \rrbracket$ from formulas to (labelled) graphs, inductively as follows:

\[
\begin{align*}
\llbracket a \rrbracket & = \bullet a \quad \text{(for any atom } a) \\
\llbracket A \lor B \rrbracket & = \llbracket A \rrbracket + \llbracket B \rrbracket \\
\llbracket A \land B \rrbracket & = \llbracket A \rrbracket \times \llbracket B \rrbracket \\
\llbracket \exists x.A \rrbracket & = x \llbracket A \rrbracket \\
\llbracket \forall x.A \rrbracket & = \bullet x \times \llbracket A \rrbracket
\end{align*}
\]

where we write $\bullet \alpha$ for a single-vertex labelled by $\alpha$.

**Example 5.** Here is the fograph of the drinker formula $\exists x (px \Rightarrow \forall y py) = \exists x (px \lor (\forall y py))$:

We define a congruence relation $\equiv$ on formulas, called equivalence, by the following equations:

\[
\begin{align*}
A \land B & \equiv B \land A \\
A \lor B & \equiv B \lor A \\
\forall x . A & \equiv \forall x . A \\
\exists x . A & \equiv \exists x . A
\end{align*}
\]

\[
\begin{align*}
\land & \equiv (A \land B) \land C \equiv A \land (B \land C) \\
\lor & \equiv (A \lor B) \lor C \equiv A \lor (B \lor C) \\
\forall . & \equiv (\forall x . A) \land (\forall y . A) \\
\exists . & \equiv (\exists x . A) \lor (\exists y . A)
\end{align*}
\]

\[
\begin{align*}
\forall x . (A \lor B) & \equiv (\forall x . A) \lor (\forall x . B) \\
\exists x . (A \land B) & \equiv (\exists x . A) \land (\exists x . B)
\end{align*}
\]

where $x$ must not be free in $B$ in the last two equations.

**Theorem 8 ([I8] §10).** Let $A, B$ be rectified formulas. Then $A \equiv B \iff \llbracket A \rrbracket = \llbracket B \rrbracket$.

**Proof.** A straightforward structural induction on formulas.

**Example 9.** $\exists x . (\exists y py \lor (\forall y py)) \equiv \exists x \forall y (py \lor py)$, and both formulas have the same (rectified) fograph $D$, below-left.

Above-center we show the cotree of the underlying cograph (illustrating the idea behind Theorem 3) and above-right is its binding graph $\vec{D}$.

**IV. First-order Combinatorial Proofs**

**A. Fonets**

Two atoms are pre-dual if they are not $t$ or $f$, and their predicate symbols are dual (e.g. $p(x, y)$ and $\overline{p}(y, z)$) and two literals are pre-dual if their labels (atoms) are pre-dual. A linked fograph $(C, \sim_C)$ is a coloured fograph $C$ such that every colour (i.e., equivalence class of $\sim_C$), called a link, consists of two pre-dual literals, and every literal is either $t$-labelled or in a link. Hence, in a linked fograph no vertex is labelled $f$.

Let $C$ be a linked fograph. The set of links can be seen as a unification problem by identifying dual predicate symbols. A dualizer of $C$ is a substitution $\delta$ unifying all the links of $C$. Since a first-order unification problem is either unsolvable or has a most general unifier, we can define the notion of most general dualizer. A dependency is a pair $(\bullet x, y)$ of an existential binder $\bullet x$ and a universal binder $y$ such that the most general dualizer assigns to $x$ a term containing $y$. A leap is either a link or a dependency. The leap graph $C^l$ of $C$ is the undirected graph $(V_C, E_C)$ where $E_C$ is the set of leaps of $C$. A vertex set $W \subseteq V_C$ induces a matching in $C$ if $W \neq \emptyset$ and for all $w \in W$, $N(w) \cap W$ is a singleton. We say that $W$ induces a bimatching in $C$ if it induces a matching in $C$ and a matching in $C^l$.

**Definition 10 ([I8] §5).** A first-order net or fonet is a linked fograph which has a dualizer but no induced bimatching.

**B. Skew Bifibrations**

A graph homomorphism $\varphi : \langle G, E_G \rangle \to \langle H, E_H \rangle$ is a fibration ([30], [31]) if for all $v \in V_G$ and $w \varphi(v) \in E_H$, there exists a unique $\overline{w} \in V_G$ such that $\overline{w}v \in E_G$ and $\varphi(\overline{w}) = w$ (indicated below-left), and is a skew fibration ([3] §3) if for all $v \in V_G$ and $w \varphi(v) \in E_H$ there exists $\overline{w} \in V_G$ such that $\overline{w}v \in E_G$ and $\varphi(\overline{w})w \not\in E_H$ (indicated below-centre). A directed graph homomorphism is a fibration if for all $v \in V_G$ and $(w, \varphi(v)) \in E_H$, there exists a unique $\overline{w} \in V_G$ such that $(\overline{w}, v) \in E_G$ and $\varphi(\overline{w}) = w$ (indicated below-right).
A fograph homomorphism $\varphi = \langle \nu, \rho_\varphi \rangle$ is a pair where $\varphi: G \to H$ is a graph homomorphism between the underlying graphs, and $\rho_\varphi$, also called the substitution induced by $\varphi$, is a variable renaming such that for all $v \in V_G$ we have $\ell(\varphi(v)) = \rho_\varphi(\ell(v))$, and $\rho_\varphi$ is the identity on variables not in $G$. Note that $\varphi$ necessarily maps binders to binders and literals to literals. Since $\rho_\varphi$ is fully determined by $\varphi$ alone, we often leave $\rho_\varphi$ implicit. A fograph homomorphism $\varphi: G \to H$ preserves existentials if for all existential binders $b$ in $G$, the binder $\varphi(b)$ is existential in $H$.

**Definition 11 ([18] §4).** Let $G$ and $H$ be fographs. A skew bifibration $\varphi: G \to H$ is an existential-preserving fograph homomorphism that is a skew fibration on $\langle V_G, E_G \rangle \to \langle V_H, E_H \rangle$ and a fibration on the binding graphs $G \to H$.

**Example 12.** Below-left is a skew bifibration, whose binding fibration is below-centre. When the labels on the source fograph can be inferred (modulo renaming), we often omit the labelling in the upper graph, as below-right.

**Definition 13 ([18] §6).** A first-order combinatorial proof (FOCP) of a fograph $G$ is a skew bifibration $\varphi: C \to G$ where $C$ is a fonet. A first-order combinatorial proof of a formula $A$ is a combinatorial proof of its graph $\llbracket A \rrbracket$.

Figure 4 shows examples of FOCPs (taken from [18]), each above the formula it proves. The same FOCPs are in Figure 5 in condensed form, with the formula graph left implicit.

**Theorem 14 ([18] §6).** FOCPs are sound and complete for first-order logic.

**Remark 15.** Our definition of FOCP is slightly more lax than the original definition of [18], as we allow for a variable renaming $\rho_\varphi$ which was restricted to be the identity in [18].

**V. First-order Deep Inference System KS1**

In contrast to standard proof formalisms, like sequent calculi or tableaux, where inference rules decompose the principal formula along its root connective, deep inference rules apply like rewriting rules inside any (positive) formula or sequent context, which is denoted by $S\{\cdot\}$, and which is a formula (resp. sequent) with exactly one occurrence of the hole $\{\cdot\}$ in the position of an atom. Then $S\{A\}$ is the result of replacing the hole $\{\cdot\}$ in $S\{\cdot\}$ with $A$.

Figure 6 shows the inference rules for the deep inference system KS1 introduced in this paper. It is a variation of the systems presented by Brünnler [32] and Ralph [33] in their...
The main differences are (i) the explicit presence of the mix-rule, (ii) a different choice of how the formula equivalence \( \equiv \) is defined, (iii) an explicit rule for the equivalence, and (iv) new inference rules \( w_y \) and \( c_y \). The reason behind these design choices is to obtain the correspondence with combinatorial proofs and the full completeness result.

We consider here only the cut-free fragment, as cut-elimination for deep inference systems has already been discussed elsewhere (e.g. [22], [34]). As with the sequent system LK1, we also need for KS1 the linear fragment, MLS1\(^x\), and that is shown in Figure 6 in the dashed box.

We write \( S \vdash \Phi \) to denote a derivation \( \Phi \) from \( B \) to \( A \) using the rules from system \( S \). A formula \( A \) is provable in a system \( S \) if there is a derivation in \( S \) from \( t \) to \( A \).

We will for some results also employ the general (non-atomic) version of the contraction rule:

\[
\begin{array}{c}
S \{ A \lor A \} \\
\hline
S \{ A \}
\end{array}
\]  

VI. MAIN RESULTS

We state the main results of this paper here, and prove them in later sections. The first is routine and expected, but must be proved nonetheless:

**Theorem 16.** KS1 is sound and complete for first-order logic.

Our second result is more surprising, as it is a very strong decomposition result for first-order logic.

**Theorem 17.** For every derivation \( KS1 \vdash \Phi \), there is \( f \)-free formulas \( A_1, \ldots, A_5 \) and a derivation

\[
\begin{array}{c}
t \\
\{ \forall, a, t \} \\
A_5 \\
\{ s, \text{mix}, t \} \\
A_4 \\
\{ \text{ac}, c, v \} \\
A_3 \\
\{ w, w, v, =, \equiv \} \\
A_1 \\
\end{array}
\]

This theorem is stronger than the existing decompositions for first-order logic, which either separate only atomic contraction and atomic weakening [32] or only contraction [33] or only the quantifiers in form of a Herbrand theorem [35], [33].

\(^x\)In the deep inference literature, the cut-free fragment is also called the down-fragment. But as we do not discuss the up-fragment here, we omit the down-arrows \( \downarrow \) in the rule names.
and atom-occurrences in the derivation corresponds exactly to the vertex-mapping induced by \( \phi \). To give an example, consider the derivation in Figure 7 which corresponds to the left-most combinatorial proof in Figures 4 and 5.

Thus, combinatorial proofs are closely related to derivations of the form (3), and since by Theorem 17 every derivation can be transformed into that form, we can say that combinatorial proofs provide a canonical proof representation for first-order logic, similarly to what proof nets are for linear logic (36).

Finally, Theorems 16, 17, and 19 imply Theorem 14 which means that we have here an alternative proof of the soundness and completeness for first-order combinatorial proofs which is simpler than the one given in [17], and improves with completeness being full (a surjection from syntactic KS1 proofs onto combinatorial proofs).

VII. TRANSLATING BETWEEN LK1 AND KS1

We prove Theorems 16, 17, and 18 mainly by translating derivations to and from the sequent calculus, and by rule permutation arguments.

A. The Linear Fragments MLL1\(^X\) and MLS1\(^X\)

We show that MLL1\(^X\) and MLS1\(^X\) are equivalent.

Lemma 20. If \( \vdash \Gamma \) is provable in MLL1\(^X\), then \( \bigvee(\Gamma) \) is provable in MLS1\(^X\).

Proof. This is a straightforward induction on the proof of \( \vdash \Gamma \) in MLL1\(^X\), making a case analysis on the bottommost rule instance. We show here only the case of \( \vdash \Delta, A \rightarrow \Delta, \forall x.A \) (all other cases are simpler or have been shown before, e.g. [32]): By induction hypothesis, there is a proof of \( \bigvee(\Delta) \lor A \) in MLS1\(^X\). We can prefix every line in that proof by \( \forall x \) and then compose the following derivation:

\[
\begin{align*}
\forall x.A &
\Downarrow
\frac{t}{\forall x.t} \\
\text{MSL1}\(^X\) &
\Downarrow
\frac{\forall x.(\forall x.A)}{\forall x.(\Delta) \lor A}
\end{align*}
\]

where we can apply the \( \Rightarrow \)-rule because \( x \) is not free in \( \Delta \).

Lemma 21. Let \( t \frac{S\{A\}}{S\{B\}} \) be an inference rule in MLS1\(^X\). Then the sequent \( \vdash \overline{A}, B \) is provable in MLL1\(^X\).

Proof. A routine exercise.

Lemma 22. Let \( A, B \) be formulas, and let \( S\{\cdot\} \) be a (positive) context. If \( \vdash \overline{A}, B \) is provable in MLL1\(^X\), then so is \( \vdash S\{A\}, S\{B\} \).

Proof. A straightforward induction on \( S\{\cdot\} \). (see e.g. [37])

Lemma 23. If a formula \( C \) is provable in MLS1\(^X\), then \( \vdash C \) is provable in MLL1\(^X\).

Proof. We proceed by induction on the number of inference steps in the proof of \( C \) in MLS1\(^X\). Consider the bottommost rule instance \( \frac{S\{A\}}{S\{B\}} \). By induction hypothesis we have a MLL1\(^X\) proof \( \Pi \) of \( \vdash S\{A\} \). By Lemmas 21 and 22, we have a MLL1\(^X\) proof of \( \vdash S\{A\}, S\{B\} \). We can compose them via the cut rule instance \( \frac{\vdash S\{A\}}{\vdash S\{B\}} \) and then apply Theorem 2.

B. Contraction and Weakening

The first observation here is that Lemmas 20 and 22 from above also hold for LK1 and KS1. We therefore immediately have:

Theorem 24. For every sequent \( \Gamma \), we have that \( \vdash \Gamma \) is provable in LK1 if and only if \( \bigvee(\Gamma) \) is provable in KS1.

Then Theorem 16 is an immediate consequence. Let us now proceed with providing further lemmas that will be needed for the other results.

Lemma 25. The c-rule is derivable in \( \{ ac, m, m_v, m_3, \equiv \} \).

Proof. This can be shown by a straightforward induction on \( A \) (for details, see e.g. [32]).

Lemma 26. \( w_v, c_v, m, m_v, m_3 \) are derivable in \( \{ w, c, \equiv \} \).

Proof. We only show the cases for \( w_v \) and \( c_v \) (for the others see [32]):

\[
\begin{align*}
\frac{A}{\forall x.((\forall x.A) \lor A)} \\
\frac{A}{(\forall x.A) \lor (\forall x.A)}
\end{align*}
\]

where in the first derivation, \( x \) is not free in \( A \), and in the second one not free in \( \forall x.A \).

Lemma 27. Let \( A \) and \( B \) be formulas. Then

\[
\begin{align*}
\frac{A}{B} &
\equiv
\frac{B}{A}
\end{align*}
\]

Proof. Immediately from Lemmas 25 and 26.

Remark 28. Observe that Lemma 27 would also hold with the rules \( w_v \) and \( c_v \) removed.

C. Rule Permutations

Theorem 29. Let \( \Gamma \) be a sequent. If \( \vdash \Gamma \) is provable in LK1 (as depicted on the left below) then there is a sequent \( \Gamma' \) not containing any \( f \), such that there is a derivation as shown on the right below:

\[
\begin{align*}
\text{LK1} &
\Downarrow
\frac{\Phi}{\vdash \bigvee(\Gamma)}
\end{align*}
\]

Proof. First, we can replace every instance of the \( f \)-rule in \( \Phi \) by \( wk \). Then the instances of \( wk \) and \( ctr \) are replaced by
Proof. First, observe that the $\exists$ rule can be permuted under all the other rules since $A[x/t]$ has the same structure as $A$ and none of the other rules has a premise of the form $S\{\exists x.A\}$. It suffices now to prove that all rules in $\{s, \text{mix}, \equiv\}$ can be permuted over the rules in $\{\forall, \alpha, \iota\}$, which is straightforward. For $s$, mix, and the $\equiv$-instances that do not involve the quantifiers, the details can be found in [38]. The $\equiv$-instances concerning the quantifiers are admissible if the $\exists$-rule is not present. \hfill $\square$

Lemma 31. For every derivation $\{w, w_v, ac, c_v, m, m_v, m_3, \equiv\}$
there are formulas $A'$ and $B'$ such that
\[
\begin{align*}
A' &\vdash A \\
\{m, m_v, m_3, \equiv\} &\vdash A' \\
\{ac, c_v\} &\vdash B' \\
\{w, w_v\} &\vdash B
\end{align*}
\]

Proof. Permute all $w$ and $w_v$ instances to the bottom of the derivation, then permute all $c$ and $c_v$ below $\{m, m_v, m_3\}$. This involves a tedious but routine case analysis. However, unlike most other rule permutations in this paper, this has not been done before in the deep inference literature. For this reason, we give the full case analysis in Appendix A. This Lemma is the reason for the presence of the rules $w_v$ and $c_v$, as without them the permutation cases in (5) could not be resolved. \hfill $\square$

We can now complete the proof of Theorems 17 and 18.

Proof of Theorem 17. Assume we have a proof of $A$ in KS1. By Theorem 24 we have a proof of $\vdash A$ in LK1 to which we can apply Theorem 29. Finally, we apply Lemma 20 to get the desired shape. \hfill $\square$

Proof of Theorem 18. Assume we have a proof of $A$ in KS1. We first apply Theorem 18 and then Lemma 30 to the upper half and Lemmas 27 and 31 to the lower half. \hfill $\square$

VIII. Fonets and Linear Proofs

A. From MLL1$^X$ Proofs to Fonets

Let $\Pi$ be a MLL1$^X$ proof of a rectified sequent $\vdash \Gamma$ not containing $\top$. We now show how $\Pi$ is translated into a linked fograph $\llbracket \Pi \rrbracket = (\llbracket \Gamma \rrbracket, \sim_\Pi)$. We proceed inductively, making a case analysis on the last rule in $\Pi$. At the same time we are constructing a dualizer $\delta_\Pi$, so that in the end we can conclude that $\llbracket \Pi \rrbracket$ is in fact a fonet.

1) $\Pi$ is $\alpha x \vdash a, \pi$: Then the only link is $\{a, \pi\}$, and $\delta_\Pi$ is empty.

2) $\Pi$ is $\top$: Then $\sim_\Pi$ and $\delta_\Pi$ are both empty.

3) The last rule in $\Pi$ is $\overline{\pi} \vdash \Gamma'$. By induction hypothesis, we have proofs $\Pi'$ and $\Pi''$ of $\Gamma'$ and $\Gamma''$, respectively. We have $\llbracket \Gamma \rrbracket = \llbracket \Gamma' \rrbracket + \llbracket \Gamma'' \rrbracket$ and we can let $\sim_\Pi = \sim_{\Pi'} \cup \sim_{\Pi''}$ and $\delta_\Pi = \delta_{\Pi'} \cup \delta_{\Pi''}$.

4) The last rule in $\Pi$ is $\overline{\pi} \vdash \Gamma$. By induction hypothesis, there is a proof $\Pi'$ of $\Gamma = \Gamma_1, A, B$. We have $\llbracket \Gamma \rrbracket = \llbracket \Gamma' \rrbracket$ and let $\sim_\Pi = \sim_{\Pi'}$ and $\delta_\Pi = \delta_{\Pi'}$.

5) The last rule in $\Pi$ is $\wedge \vdash \Gamma_1, A \wedge B, \Gamma_2$. By induction hypothesis, we have proofs $\Pi'$ and $\Pi''$ of $\Gamma' = \Gamma_1, A$ and $\Gamma'' = B, \Gamma_2$, respectively. We have $\llbracket \Gamma \rrbracket = \llbracket \Gamma' \rrbracket + (\llbracket A \rrbracket \times \llbracket B \rrbracket) + \llbracket \Pi'' \rrbracket$ and we let $\sim_\Pi = \sim_{\Pi'} \cup \sim_{\Pi''}$ and $\delta_\Pi = \delta_{\Pi'} \cup \delta_{\Pi''}$.

6) The last rule in $\Pi$ is $\exists \vdash \Gamma_1, A[x/t]$. By induction hypothesis, there is a proof $\Pi'$ of $\Gamma' = \Gamma_1, A[x/t]$. For each atom in $\Gamma' = \Gamma_1, A[x/t]$, there is a corresponding atom in $\Gamma = \Gamma_1, \exists x. A$. We can therefore define the linking $\sim_\Pi$ from the linking $\sim_{\Pi'}$ via this correspondence. Then, we let $\delta_\Pi$ be $\delta_{\Pi'} + [x/t]$. Since $\Gamma$ is rectified $x$ does not yet occur in $\delta_{\Pi'}$. Hence $\delta_\Pi$ is a dualizer of $\llbracket \Pi \rrbracket$.

7) The last rule in $\Pi$ is $\forall \vdash \Gamma_1, A (x$ not free in $\Gamma_1)$

By induction hypothesis, there is a proof $\Pi'$ of $\Gamma' = \Gamma_1, A$, which has the same atoms as in $\Gamma = \Gamma_1, \forall x. A$. Hence, we can let $\sim_\Pi = \sim_{\Pi'}$ and $\delta_\Pi = \delta_{\Pi'}$. \hfill $\square$

Theorem 32. If $\Pi$ is a MLL1$^X$ proof of a rectified $\top$-free sequent $\vdash \Gamma$, then $\llbracket \Pi \rrbracket$ is a fonet and $\delta_\Pi$ a dualizer for it.

Proof. We must show that none of the operations above introduces a bimatching. For cases 1–6, this is immediate. For case 7, observe that there is a potential dependency from each existential binder in $\llbracket \Gamma' \rrbracket$ to the new $x$-binder $\exists x$ in $\llbracket \Gamma \rrbracket$. However, observe that this $\exists x$ vertex is not connected to any vertex in $\llbracket \Gamma' \rrbracket$, and hence no such new dependency can be extended to a bimatching. That $\delta_\Pi$ is a dualizer for $\llbracket \Pi \rrbracket$ follows immediately from the construction. Hence, $\llbracket \Pi \rrbracket$ is a fonet. \hfill $\square$
**B. From MLS**$^1$**$^X$ Proofs to Fonets**

There is a more direct path from a ML$^1$$^l$$^l$ proof $\Pi$ of a rectified sequent $\Gamma$ to the linked fograph $[\Pi]$; take the fograph $[[\Gamma]]$, and let the equivalence classes of $\sim_{\Pi}$ be all the atom pairs that meet in an instance of $\alpha$, and $\delta_{\Pi}$ comprises the substitutions at the $\exists$-rules in $\Pi$. We chose the more cumbersome path above because it gives us a direct proof of Theorem 52. However, for translating ML$^1$$^X$ derivation into fonets, we employ exactly that direct path.

In a derivation in ML$^1$$^X$ where the conclusion is rectified, every line is also rectified, as the only rules involving bound variables are $\forall$ and $\exists$ which (upwards) both remove a binder. Therefore, we can call such a derivation 

**rectified**, and for a non-rectified ML$^1$$^X$ derivation $\Phi$ we can define its rectification $\hat{\Phi}$ inductively, by rectifying each line, proceeding step-wise from conclusion to premise $^3$

A rectified derivation ML$^1$$^X$ $\Phi$ determines a substitution $t$ which maps the existential bound variables occurring in $A$ to the terms substituted for them in the instances of the $\exists$-rule in $\Phi$. We denote this substitution by $\delta_\Phi$ and call it the dualizer of $\Phi$. Furthermore, every atom occurring in the conclusion $A$ must be consumed by a unique instance of the rule $\exists a_i$. This allows us to define a (partial) equivalence relation $\sim_{\Phi}$ on the atom occurrences in $A$ by $a_i \sim_{\Phi} b$ if $a$ and $b$ are consumed by the same instance of $a_i$ in $\Phi$. We call $\sim_{\Phi}$ the linking of $\Phi$, and define $[\Phi] = ([A], \sim_{\Phi})$.

**Theorem 33.** Let ML$^1$$^X$ $\Phi$ be a rectified derivation where $A$ is $f$-free. Then $[\Phi]$ is a fonet and $\delta_\Phi$ a dualizer for it.

To prove this theorem, we have to show that no inference rule in ML$^1$$^X$ can introduce a bimatching. To simplify the proof, we introduce the frame $\Psi_1$ of the linked fograph $C$, which is a linked (propositional) cograph in which the dependencies between the binders in $C$ are encoded as links.

More formally, let $C$ be a formula with $[C] = C$, to which we exhaustively apply the following subformula rewriting steps, to obtain a sequent $C^*$:

1. **Encode dependencies as fresh links.** For each dependency $\{x_i,y_j\}$ in $C$, with corresponding subformulas $\exists x_i.A$ and $\forall y_j.B$ in $C$, we pick a fresh (nullary) predicate symbol $q_{i,j}$, and then replace $\exists x_i.A$ by $\psi_{i,j} \land \exists x_i.A$, and replace $\forall y_j.B$ by $\psi_{i,j} \lor \forall y_j.B$.
2. **Erase quantifiers.** After step 1, remove all the quantifiers, i.e., replace $\exists x_i.A$ by $A$ and replace $\forall y_j.B$ by $B$ everywhere.
3. **Simplify atoms.** After step 2, replace every predicate $p_1 \ldots p_n$ (resp. $\overline{p}_1 \ldots \overline{p}_n$) with a nullary predicate symbol $p$ (resp. $\overline{p}$).

Then $\sim_{C^*}$ consists of the pairs induced by $\sim_C$ and the new pairs $\{q_{i,j},\overline{q}_{i,j}\}$ introduced in step 1 above. We call $C^*$ the frame of $C$ and we define the frame of $\Phi$, denoted $C^*_\Phi$, as $[[C^*]], \sim_{C^*}$.

**Lemma 34.** If a linked fograph $C$ has an induced bimatching then so does its frame $C^*$.

**Proof.** Immediately from the construction of the frame. □

**Proof of Theorem 33.** From $\Phi$ we construct a derivation $\Phi^*$ of $A^*$ in the propositional fragment of ML$^1$$^X$, such that $[[\Phi^*]] = [\Phi]^*$. The rules $\exists a_i, t, \mix$ and $\mix$ are translated trivially, and for $\equiv$, it suffices to observe that the frame construction is invariant under $\equiv$. Finally, for the rules $\forall$ and $\exists$, proceed as follows. Every instance of $\forall$ is replaced by the derivation on the right below.

$$\forall S[t] S[\forall y_j] \rightsquigarrow S[\{q_{h_1,j} \lor \psi_{h_1,j} \land \ldots \land \{q_{h_n,j} \lor \psi_{h_n,j} \land \land t\}\} \equiv, \exists] \equiv S[q_{h_1,j} \lor \ldots \lor q_{h_n,j} \lor (\psi_{h_1,j} \land \ldots \land \psi_{h_n,j}) \land t]$$

where $h_1, \ldots, h_n$ range over the indices of the existential binders dependent on that $y_j$. It is easy to see how $\forall \equiv, \exists$ is constructed. The construction of $\Psi_2$, using $\forall$ and $\exists$, is standard, see, e.g. 40, 47, 41, 48. Then, every occurrence of $\forall y_j, F$ is replaced by $q_{h_{ij},j} \lor \ldots \lor q_{h_{kn},j} \lor (\psi_{h_{ij},j} \land \ldots \land \psi_{h_{kn},j} \lor F')$ in the derivation below that $\forall$-instance. Now, observe that all instances of the $\exists$-rule introducing $x_i$ dependent on $y_j$ must occur below in the derivation (otherwise $\Phi$ would not be rectified). Now consider such an instance $\exists S[B[x_i/t]]$.

Its context $S[\cdot]$ must contain all the $\forall y_j$ the $\exists x_i$ depends on, such that $B$ is in its scope. Following the translation of the $\forall$ rules above, we can therefore translate the $\exists$-rule instance by the following derivation.

$$S_0[S_{i,k_1} \land S_1[S_{i,k_2} \land \ldots \land S_{i-1}[S_{i,k_i} \land S_i(B') \ldots ] \ldots ] \equiv, \exists] \equiv S_0[S_1[S_2[S_3[S_4[S_{i,k_1} \land S_{i,k_2} \land \ldots \land q_{i,k_1} \land B'] \ldots ] \ldots ]$$

where $k_1, \ldots, k_l$ are the indices of the universal binders on which $x_i$ depends, and $B'$ is in which all predicates are replaced by a nullary one (step 3 in the frame construction). The derivation $\Psi_3$ can be constructed in the same way as $\Psi_2$.

Doing this to all instances of the rules $\forall$ and $\exists$ in $\Phi$ yields indeed a propositional derivation $\Phi^*$ with $[[\Phi^*]] = [\Phi]^*$. It has been shown by Retoré [42] and rediscovered by Straßburger [38] that $[[\Phi^*]] = ([C^*]_\Phi, \sim_{C^*})$ cannot contain an induced bimatching. By Lemma 33 $[\Phi]$ does not have an induced bimatching either. Furthermore, it follows from the definition of $\delta_\Phi$ that it is a dualizer for $[[\Phi]]$.

**Remark 35.** There is an alternative path of proving Theorem 33 by translating $\Phi$ to an ML$^1$$^X$-proof $\Pi$, observing that this process preserves the linking and the dualizer. However,

---

$^3$As for formulas, the rectification of a derivation is unique up to renaming of bound variables.

$^4$For better readability we omit superfluous parentheses, knowing that we always have $\equiv$ incorporating associativity and commutativity of $\land$ and $\lor$. 

9
for this, we have to extend the construction from the previous subsection to the cut-rule, and then show that linking and dualizer of a sequent proof \( \Pi \) are invariant under cut elimination. This can be done similarly to unification nets in [39].

C. From Fonets to MLL\(^1\)\(^X\) Proofs

Now we are going to show how from a given fonet \( \langle C, \sim_C \rangle \) we can construct a sequent proof \( \Pi \) in MLL\(^1\)\(^X\) such that \( \Pi \) = \( \langle C, \sim_C \rangle \). In the proof net literature, this operation is also called *sequentialization*. The basic idea behind our sequentialization is to use the frame of \( C \), to which we can apply the splitting tensor theorem, and then reconstruct the sequent proof \( \Pi \).

Let \( \Gamma \) be a propositional sequent and \( \sim \Gamma \) be a linking for \( \langle \Gamma \rangle \). A conjunction formula \( A \land B \) is splitting or a splitting tensor if \( \Gamma = \Gamma', A \land B, \Gamma'' \) and \( \sim \Gamma = \sim \Gamma_1 \sim \Gamma_2 \), such that \( \sim \Gamma_1 \) is a linking for \( \langle \Gamma' \rangle \), and \( \sim \Gamma_2 \) is a linking for \( \langle B, \Gamma'' \rangle \), i.e., removing the \( \land \) from \( A \land B \) splits the linked fograph \( \langle \Gamma \rangle \sim \Gamma \) into two fographs. We say that \( \langle \Gamma' \rangle \sim \Gamma_1 \) is mixed \( \iff \Gamma = \Gamma', \Gamma'' \) and \( \sim \Gamma_1 = \sim \Gamma_1 \cup \sim \Gamma_2 \), such that \( \sim \Gamma_2 \) is a linking for \( \langle B \rangle \) and \( \sim \Gamma_2 \) is a linking for \( \langle \Gamma'' \rangle \). Finally, \( \langle \Gamma \rangle \sim \Gamma \) is splittable if it is mixed or has a splitting tensor.

**Theorem 36.** Let \( \Gamma \) be a \( \beta \)-free propositional sequent containing only atoms and \( \land \)-formulas, and \( \sim \Gamma \) be a linking for \( \langle \Gamma \rangle \). If \( \langle \Gamma \rangle \sim \Gamma \) does not induce a bimatching then it is splittable.

This is the well-known splitting-tensor-theorem [19], [43], adapted for the presence of mix. In the setting of linked cographs, it has first been proved by Retoré [44], [45] and then rediscovered by Hughes [9]. We use it now for our sequentialization:

**Theorem 37.** Let \( \langle C, \sim_C \rangle \) be a fonet, and let \( \Gamma \) be a sequent with \( \Gamma = C \). Then there is an MLL\(^1\)\(^X\)-proof \( \Pi \) of \( \Gamma \), such that \( \Pi = \langle C, \sim_C \rangle \).

**Proof.** Let \( \delta_C \) be the dualizer of \( \langle C, \sim_C \rangle \). We proceed by induction on the size of \( \Gamma \) (i.e., the number of symbols in it, without counting the commas). If \( \Gamma \) contains a formula with \( \lor \)-root, or a formula \( \forall x. A \), we can immediately apply the \( \lor \)-rule of \( \forall \)-rule of MLL\(^1\)\(^X\) and proceed by induction hypothesis. If \( \Gamma \) contains a formula \( \exists x. A \) such that the corresponding binder \( \bullet x \) in \( C \) has no dependency, then we can apply the \( \exists \)-rule, choosing the term \( t \) as determined by \( \delta_C \), and proceed by induction hypothesis. Hence, we can now assume that \( \Gamma \) contains only atoms, \( \land \)-formulas, or formulas of shape \( \exists x. A \), where the vertex \( \bullet x \) has dependencies. Then the frame \( \langle \langle \Gamma' \rangle \sim \Gamma \rangle \) does not induce a bimatching and contains only atoms and \( \land \)-formulas, and is therefore splittable. If it is mixed, then we can apply the \( \land \)-rule to \( \Gamma \) and apply the induction hypothesis to the two components. If it is not mixed then there must be a splitting tensor. If the splitting \( \land \) is already in \( \Gamma \), then we can apply the \( \land \)-rule and proceed by induction hypothesis on the two branches. However, if \( \Gamma' \) is not mixed and all splitting tensors are \( \land \)-formulas introduced in step 1 of the frame construction, then we get a contradiction as in that case there must be a \( \lor \) or \( \forall \)-formula in \( \Gamma \).

D. From Fonets to MLL\(^1\)\(^X\) Proofs

We can now straightforwardly obtain the same result for MLL\(^1\)\(^X\).

**Theorem 38.** Let \( \langle C, \sim_C \rangle \) be a fonet, and let \( \Gamma \) be a formula with \( \Gamma = C \). Then there is a derivation MLL\(^1\)\(^X\) \( \Phi \) such that \( \Phi = \langle C, \sim_C \rangle \).

**Proof.** We apply Theorem 37 to obtain a sequent proof \( \Pi \) of \( \Gamma \) with \( \Pi = \langle C, \sim_C \rangle \). Then we apply Lemma 20 observing that the translation from MLL\(^1\)\(^X\) to MLL\(^1\)\(^X\) preserves linking and dualizer. \( \square \)

**Remark 39.** Note that it is also possible to do a direct “sequentialization” into the deep inference system MLL\(^1\)\(^X\), using the techniques presented in [38] and [46].

IX. Skew Bifibrations and Resource Management

In this section we establish the relation between skew bifibrations and derivations in \( \{ w, w_v, a_c, c_v, m_v, m_3, \equiv \} \). However, if a derivation \( \Phi \) contains instances of the rules \( c_v \), \( m_v \), and \( m_3 \) we can no longer naively define the rectification \( \Phi \) as in the previous section for MLL\(^1\)\(^X\), as these two rules cannot be applied if premise and conclusion are rectified. For this reason we define here rectified versions \( \hat{c}_v \), \( \hat{m}_v \), and \( \hat{m}_3 \), shown below:

\[
\hat{c}_v \cdot S\{ \forall y. A y \} \lor \forall z. B z \} \\
\hat{m}_v \cdot S\{ \forall x. (A x \lor B x) \} \\
\hat{m}_3 \cdot S\{ \exists z. (A x \lor B x) \}
\]

Here, we use the notation \( A \cdot \) for a formula \( A \) with occurrences of a placeholder \( \cdot \) for a variable. Then \( Ax \) stands for the results of replacing that placeholder with \( x \), and also indicating that \( x \) must not occur in \( A \). Then \( \forall x. Ax \) and \( \forall y. Ay \) are the same formula modulo renaming of the bound variable bound by the outermost \( \forall \)-quantifier. We also demand that the variables \( x \), \( y \), and \( z \) do not occur in the context \( S\{ \cdot \} \).

Note that in an instance of \( \hat{m}_v \) or \( \hat{m}_3 \) (as shown above), we can have \( x = y \) or \( x = z \), but not both if the premise is rectified. If \( x = y \) and \( x = z \) we have \( m_v \) and \( m_3 \) as special cases of \( \hat{m}_v \) and \( \hat{m}_3 \), respectively. And similarly, if \( x = y \) then \( c_v \) is a special case of \( \hat{c}_v \).

For a derivation \( \Phi \) in \( \{ w, w_v, a_c, c_v, m_v, m_3, \equiv \} \), we can now construct the rectification \( \Phi \) by rectifying each line of \( \Phi \), yielding a derivation in \( \{ w, w_v, a_c, \hat{c}_v, m_v, \hat{m}_v, m_3, \equiv \} \).

For each instance \( \tau \frac{Q}{P} \) of an inference rule in \( \{ w, w_v, a_c, \hat{c}_v, m_v, \hat{m}_v, m_3, \equiv \} \) we can define the *induced map* \( \tau \frac{Q}{P} \) which acts as the identity for \( r \in \{ m, \equiv \} \) and as the canonical injection for \( r \in \{ w, w_v \} \). For \( r = ac \) it maps the vertices corresponding to the two atoms in the premise to the vertex of the contracted atom in the conclusion, and for \( r \in \{ \hat{c}_v, \hat{m}_v, \hat{m}_3 \} \) it maps the two
vertices corresponding to the quantifiers in the premise to the one in the conclusion (and acts as the identity on all other vertices). For a derivation \( \Phi \) in \( \{w, w', a, c, v, m, \overline{m}_V, \overline{m}_3, \equiv\} \) we can then define the **induced map** \( \Phi \) as the composition of the induced maps of the rule instances in \( \Phi \).

Lemma 40. Let \( \{w, w', a, c, v, m, \overline{m}_V, \overline{m}_3, \equiv\} \) be given. Then there is a rectified derivation \( \{w, w', a, c, v, m, \overline{m}_V, \overline{m}_3, \equiv\} \to B \) such that the induced maps \( \Phi : \llbracket A \rrbracket \to \llbracket B \rrbracket \) and \( \hat{\Phi} : \llbracket A \rrbracket \to \llbracket B \rrbracket \) are equal up to a variable renaming of the vertex labels.

Proof. Immediate from the definition. \( \square \)

A. From Contraction and Weakening to Skew Bifibrations

Lemma 41. Let \( \{w, w', a, c, v, m, \overline{m}_V, \overline{m}_3, \equiv\} \) be a rectified derivation. Then the induced map \( \Phi : \llbracket A \rrbracket \to \llbracket B \rrbracket \) is a skew bifibration.

Before we show the proof of this lemma, we introduce another useful concept: the **propositional encoding** \( A^\circ \) of a formula \( A \), which is a propositional formula with the property that \( \llbracket A \rrbracket = \llbracket A^\circ \rrbracket \). For this, we introduce new propositional variables that have the same names as the (first-order) variables \( x \in \text{VAR} \). Then \( A^\circ \) is defined inductively by:

\[
\begin{align*}
A^\circ &= a \\
(A \lor B)^\circ &= A^\circ \lor B^\circ \\
(A \land B)^\circ &= A^\circ \land B^\circ \\
(\forall x A)^\circ &= x \lor A^\circ \\
(\exists x A)^\circ &= x \land A^\circ 
\end{align*}
\]

Lemma 42. For every formula \( A \), we have \( \llbracket A^\circ \rrbracket = \llbracket A \rrbracket \).

Proof. A straightforward induction on \( A \). \( \square \)

Proving Lemma 41. First, observe that for every inference rule \( r \in \{w, w', a, c, v, m, \overline{m}_V, \overline{m}_3, \equiv\} \) the induced map \( r^\circ : \llbracket Q \rrbracket \to \llbracket P \rrbracket \) defines an existential-preserving graph homomorphism \( \llbracket Q \rrbracket \to \llbracket P \rrbracket \) and a fibration on the corresponding binding graphs. Therefore, their composition \( \Phi \) has the same properties of fibration.

For showing that it is also a skew fibration, we construct for \( \Phi \) its propositional encoding \( \hat{\Phi}^\circ \) by translating every line into its propositional encoding. The instances of the rules \( m_v \) and \( m_3 \) are replaced by:

\[
\begin{align*}
S\{x \lor (\forall y A)^\circ \} &\equiv \overline{a}c \\
S\{x \lor (\forall y A)^\circ \lor (B z)^\circ \} &\equiv \overline{a}c \\
S\{x \lor ((Ax)^\circ \lor (B z)^\circ) \} &\equiv \overline{a}c \\
S\{x \lor ((Ax)^\circ \lor (B z)^\circ) \} &\equiv \overline{m}
\end{align*}
\]

respectively, where \( \overline{a}c \) is a \( c \) that renames the variables—the propositional variable, as well as the first-order variable of the same name—as everything is rectified, there is no ambiguity here. Any instance of a rule \( w \), \( ac \), \( m \), or \( \equiv \) is translated to an instance of the same rule, \( \overline{c}v \) is translated to \( \overline{a}c \), and \( w_v \) is translated to \( \overline{w}v \).

This gives us a derivation \( \{w, ac, \overline{a}c, m, \overline{m}_V, \overline{m}_3, \equiv\} \to B^\circ \) such that \( \llbracket \Phi^\circ \rrbracket = \llbracket \Phi \rrbracket \). It has been shown in [23] that \( \llbracket \Phi^\circ \rrbracket \) is a skew fibration. Hence, \( \Phi \) is a skew fibration. \( \square \)

B. From Skew Bifibrations to Contraction and Weakening

Lemma 43. Let \( A \) and \( B \) be graphs, let \( \varphi : A \to B \) be a skew bifibration, and let \( A \) and \( B \) be formulas with \( \llbracket A \rrbracket = A \) and \( \llbracket B \rrbracket = B \). Then there are derivations

\[
\begin{align*}
A &\to \Phi \to B \to B^\circ \quad \text{and} \quad \{w, w', a, c, v, m, \overline{m}_V, \overline{m}_3, \equiv\} \to B^\circ\Phi \\
\end{align*}
\]

such that \( \llbracket \Phi \rrbracket = \varphi \) and \( \hat{\Phi} \) is a rectification of \( \Phi \), and \( \rho \varphi \) is the substitution induced by \( \varphi \).

In the proof of this lemma, we make use of the following concept: Let \( \Psi \) be a derivation where \( P \) and \( Q \) are propositional formulas (possibly using variable \( x \in \text{VAR} \) at the places of atoms). We say that \( \Psi \) can be lifted to \( \Psi' \) if there are (first-order) formulas \( C \) and \( D \) such that \( P = C^\circ \) and \( Q = D^\circ \) and there is a derivation \( \{w, w', \equiv\} \to \Psi' \).

We say a fograph homomorphism \( \varphi : G \to H \) is full if for all \( v, w \in V_G \), we have that \( \varphi(v) \varphi(w) \in E_H \) implies \( vw \in E_G \).

Lemma 44. Let \( \varphi : G \to H \) be full and injective skew bifibration such that \( \rho \varphi \) is the identity substitution, and let \( G \) and \( H \) be formulas with \( \llbracket G \rrbracket = G \) and \( \llbracket H \rrbracket = H \). Then there is a derivation \( \{w, w', \equiv\} \to \Psi \).

Proof. By [23] Proposition 7.6.1], we have a derivation \( \{w, w', \equiv\} \to \Psi \).

In order to lift \( \Psi \), we need to reorganize the instances of \( w \). If \( H \) contains a subformula \( \forall x.A \) which is not present in \( G \), the \( w \)-instances in \( \Psi \) could introduce the parts of the propositional encoding \( x \lor A \) independently. We say that an instance \( r_1 \) of \( w \) in \( \Psi \) is in the scope of an instance \( r_2 \) of \( w \) if \( r_1 \) introduced formulas that contain a free variable \( x \) (i.e., \( x \) occurs in a term in a predicate) and \( r_2 \) introduces the atom \( x \) as a subformula (i.e., the propositional encoding of the binder \( \forall \) ). We can now permute the \( w \)-instances in \( \Psi \) such that whenever a rule instance \( r_1 \) is in the scope of an instance \( r_2 \), then \( r_2 \) occurs below \( r_1 \) in \( \Psi \). Then we can lift \( \Psi \) stepwise. First, observe that each line of \( \Psi \) is \( \equiv \)-equivalent to the propositional encoding \( P^\circ \) of a first-order formula \( P \).

We now have to show that each instance of \( w \) in \( \Psi \) is indeed
the image of a correct application of \( \psi \) or \( \psi \varepsilon \) in first-order logic. If we have a \( \psi \) of the form

\[
\frac{S^0 \{ A^0 \}}{w} \quad \text{or} \quad \frac{S^0 \{ (x \vee B^0) \wedge A^0 \}}{w}
\]

then \( x \) cannot occur freely in \( A \), as otherwise the fibration property would be violated. We can therefore lift these instances to

\[
\frac{S \{ A \}}{w \psi \varepsilon} \quad \text{or} \quad \frac{S \{ (\forall x. B) \wedge A \}}{w \psi \varepsilon}
\]

respectively. If a weakening happens inside a subformula \( x \vee C^0 \) or \( x \wedge C^0 \) in \( \Psi \), then there are the following cases:

\[
\frac{S^0 \{ x \vee C^0 \}}{w} \quad \frac{S^0 \{ x \wedge C^0 \}}{w} \quad \frac{S^0 \{ (x \vee D^0) \wedge C^0 \}}{w}
\]

The first two cases can be lifted to

\[
\frac{S \{ \forall x. C \}}{w} \quad \text{and} \quad \frac{S \{ \exists x. C \}}{w}
\]

respectively. But in the third case, an \( \exists \)-quantifier would be transformed into an \( \forall \)-quantifier. But as \( \psi \) to have preserve existentials, this third case cannot occur. All other situations can be lifted trivially, giving us \( \{ w, w \varepsilon, \varepsilon \} \frac{\phi}{H} \) as desired.

**Lemma 45.** Let \( \phi : \mathcal{G} \to \mathcal{H} \) be a surjective skew bifibration, and let \( G \) and \( H \) be formulas with \( \llbracket G \rrbracket = \mathcal{G} \) and \( \llbracket H \rrbracket = \mathcal{H} \). Then there is a derivation

\[
G \rho_\phi \quad \{ ac, cv, m, m_\varepsilon, m_3, \varepsilon \} \frac{\phi}{H}
\]

where \( \rho_\phi \) is the substitution induced by \( \phi \).

**Proof.** By \cite{47} Proposition 7.5, there is a derivation

\[
(G \rho_\phi)^0 \quad \{ ac, m, \varepsilon \} \frac{\psi}{\Psi}.
\]

We can lift \( \psi \) to a first-order derivation in

\[
H^0 \quad \{ ac, cv, m, m_\varepsilon, m_3, \varepsilon \},
\]

in a similar way as in the previous lemma. The technical details are in Appendix C.

**Proof of Lemma 43** Let \( V_B^o \subseteq V_B^o \) be the image of \( \phi \), and let \( B_1 \) be the subgraph of \( B \) induced by \( V_B^o \). Hence, we have two maps \( \phi'' : A \to B_1 \) being a surjection and \( \phi' : B_1 \to B \) being a full injection. Both, \( \phi' \) and \( \phi'' \) remain skew bifibrations. Furthermore, \( B_1 \) is also a fograph. Let \( B_1 \) be a formula with \( \llbracket B_1 \rrbracket = B_1 \). We can apply Lemmas 44 and 45 to obtain derivations

\[
\frac{B_1 \rho_{\phi''}}{B} \quad \text{and} \quad \frac{A \rho_{\phi''}}{B_1}
\]

As \( \rho_{\phi''} \) is the identity, we have \( \rho_{\phi''} = \rho_{\phi'} \). Hence, the composition of \( \Phi'' \) and \( \Phi' \) is the desired derivation \( \Phi \). Then \( \Phi \) can be constructed by rectifying \( \Phi' \), where the variables to be used in \( A \) are already given. That \( \phi = [\Phi] \) follows immediately from the construction.

X. **Summary and Proof of Main Result**

The only theorem of Section VII that has not yet been proved is Theorem 19 establishing the full correspondence between decomposed proofs in KS1 and combinatorial proofs. We show the proof here, by summarizing the results of the previous two Sections VIII and IX.

**Proof of Theorem 19** First, assume we have a combinatorial proof \( \phi : C \to A \) and a formula \( A \) with \( A = [A] \). Let \( B \) be a formula with \( [B] = C \), and let \( \rho_\phi \) be the substitution induced by \( \phi \). By Lemma 43 there is a derivation

\[
C \rho_\phi \quad \{ w, w \varepsilon, ac, cv, m, m_\varepsilon, m_3, \varepsilon \} \frac{\phi_2}{A}
\]

Since \( C \) is a fonet, we have by Theorem 38 a derivation

\[
t \quad \text{MLSI}^1 \left[ \frac{\phi_1}{C} \right]
\]

This derivation remains valid if we apply the substitution \( \rho_\phi \) to every line in \( \phi_1 \), yielding the derivation \( \phi_1 \) of \( C \rho_\phi \) as desired. Conversely, assume we have a decomposed derivation

\[
t \quad \text{MLSI}^1 \left[ \frac{\phi_1}{A'} \right]
\]

Then we can transform \( \phi_1 \) into a rectified form \( \hat{\phi}_1 \), proving \( A' \). By Theorem 33 the linked fograph \( \llbracket \hat{\phi}_1 \rrbracket = ([A'], \sim_{\hat{\phi}_1}) \) is a fonet. Then, by Lemma 40 there is a rectified derivation

\[
\{ w, w \varepsilon, ac, cv, m, m_\varepsilon, m_3, \varepsilon \} \frac{\phi_2}{A} \quad \text{whose induced map } \hat{\phi}_2 : [A'] \to [A]
\]

\([\hat{A}] \) is the same as the induced map \( \phi_2 : [A'] \to [A] \) of \( \phi_2 \). By Lemma 41 this map is a skew bifibration. Hence, we have a combinatorial proof \( \phi : C \to [A] \) with \( C = [A] \).

Note that Theorem 19 shows at the same time soundness, completeness, and full completeness, as

1) every proof in KS1 can be translated into a combinatorial proof, and

2) every combinatorial proof is the image of a KS1-proof under that translation.

XI. **Conclusion**

We uncovered a close correspondence between first-order combinatorial proofs and decomposed deep inference derivations of system KS1, and showed that every proof in KS1 has such a decomposed form.

The most surprising discovery for us was that all technical difficulties in our work could be reduced (in a non-trivial way) to the propositional setting.
The obvious next step in our research is to investigate proof composition and normalisation of first-order combinatorial proofs. Even in the propositional setting, the normalisation of combinatorial proofs is underdeveloped. There exist two different procedures for cut elimination for combinatorial proofs in classical propositional logic [10], [12], but both have their insufficiencies, and have not been extended to other logics.

We hope to garner new insights on the normalisation of classical first-order proofs through our work on combinatorial proofs.

REFERENCES

[1] G. Frege, Begriffsschrift. Louis Nebert, Halle, 1879, English Translation in: J. van Heijenoort (ed.), From Frege to Gödel, Harvard University Press: 1977.
[2] D. Hilbert, “Die logischen Grundlagen der Mathematik,” Mathematische Annalen, vol. 88, pp. 151–165, 1922.
[3] G. Gentzen, “Untersuchungen über das logische Schließen. I. Mathematische Zeitschrift, vol. 39, pp. 176–210, 1935.
[4] G. Gentzen, “Untersuchungen über das logische Schließen. II. Mathematische Zeitschrift, vol. 39, pp. 405–431, 1935.
[5] R. M. Smullyan, First-Order Logic. Berlin: Springer-Verlag, 1968.
[6] J. A. Robinson, “A Machine-Orientation Logic Based on the Resolution Principle,” Journal of the ACM, vol. 12, pp. 23–41, 1965.
[7] R. Thiele, “Hilbert’s Twenty-fourth Problem,” American Mathematical Monthly, vol. 110, pp. 1–24, 2003.
[8] D. Hilbert, “Mathematische Probleme,” Nachrichten der Königlichen Gesellschaft der Wissenschaften zu Göttingen, mathematisch-physikalische Klasse, vol. 3, pp. 253–297, 1900.
[9] D. Hughes, “Proofs Without Syntax,” Annals of Mathematics, vol. 164, no. 3, pp. 1065–1076, 2006.
[10] D. Hughes, “Towards Hilbert’s 24th Problem: Combinatorial Proof Invariants (preliminary version),” Electronic Notes in Theoretical Computer Science, vol. 165, pp. 37–63, 2006.
[11] L. Straßburger, “The Problem of Proof Identity, and Why Computer Scientists Should Care About Hilbert’s 24th Problem,” Philosophical Transactions of the Royal Society A, vol. 377, no. 2140, p. 20180038, 2018.
[12] L. Straßburger, “Combinatorial Flows and Their Normalisation,” in 2nd International Conference on Formal Structures for Computation and Deduction (FSCD 2017). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.
[13] L. Straßburger, “Combinatorial Flows and Proof Compression,” Inria Saclay, Research Report RR-9048, 2017. [Online]. Available: https://hal.inria.fr/hal-01498468
[14] M. Acciavio and L. Straßburger, “From Syntactic Proofs to Combinatorial Proofs,” in Automated Reasoning - 9th International Joint Conference, IJCAR 2018. Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 14-17, 2018, Proceedings, D. Galmiche, S. Schulz, and R. Sebastiani, Eds., vol. 10900. Springer, 2018, pp. 481–497.
[15] M. Acciavio and L. Straßburger, “On Combinatorial Proofs for Logics of Relevance and Entailment,” in 26th Workshop on Logic, Language, Information and Computation (WoLLIC 2019), R. Jemhoff and M. Moortgat, Eds. Springer, 2019.
[16] M. Acciavio and L. Straßburger, “On Combinatorial Proofs for Modal Logic,” in International Conference on Automated Reasoning with Analytic Tableaux and Related Methods. Springer, 2019, pp. 223–240.
[17] W. Heijltjes, D. Hughes, and L. Straßburger, “Intuitionistic Proofs Without Syntax,” in 2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS). IEEE, 2019, pp. 1–13.
[18] D. Hughes, “First-order Proofs Without Syntax,” arXiv preprint arXiv:1906.11236, 2019.
[19] J.-Y. Girard, “Linear Logic,” Theoretical Computer Science, vol. 50, pp. 1–102, 1987.
[20] S. Abramsky and R. Jagadeesan, “Games and Full Completeness for Multiplicative Linear Logic,” Journal of Symbolic Logic, vol. 59, no. 2, pp. 543–574, 1994.
[21] J. Herbrand, “Recherches sur la Théorie de la Démonstration,” Ph.D. dissertation, University of Paris, 1930.
[22] K. Brünnler, “Cut Elimination Inside a Deep Inference System for Classical Predicate Logic,” Studia Logica, vol. 82, no. 1, pp. 51–71, 2006.
[23] L. Straßburger, “A Characterization of Medial as Rewriting Rule,” in International Conference on Rewriting Techniques and Applications. Springer, 2007, pp. 344–358.
[24] A. S. Troelstra and H. Schwichtenberg, Basic Proof Theory. Cambridge University Press, 2000, no. 43.
[25] J.-Y. Girard, “Quantifiers in Linear Logic,” Temi e prospettive della logica e della filosofia della scienza contemporanea, vol. 1, pp. 95–130, 1988.
[26] A. Fleury and C. Retoré, “The Mix Rule,” Math. Structures in Comp. Science, vol. 4, no. 2, pp. 273–285, 1994.
[27] G. Bellin, “Subnets of Proof-nets in Multiplicative Linear Logic with MIX,” Mathematical Structures in Computer Science, vol. 7, no. 6, pp. 663–699, 1997.
[28] H. Lecrh, “On cliques and kernels,” 1971, Tech. report, U. Toronto.
[29] D. G. Corneil, H. Lecrh, and L. K. Stewart-Burlingham. “Complement reducible graphs,” Disc. Appl. Math., 1981.
[30] A. Grothendieck, Technique de descente et théorèmes d’existence en géométrie algébrique. I. Généralités. Descente par morphismes fidèlement plats,” in Séminaire Bourbaki: années 1958/59-1959/60, exposés 169–204. Société mathématique de France, 1960.
[31] J. W. Gray, “Fibred and cofibred categories,” in Proc. Conf. on Categorical Algebra ’65. Springer, 1966, pp. 21–83.
[32] K. Brünnler, “Deep Inference and Symmetry for Classical Proofs,” Ph.D. dissertation, Technische Universität Dresden, 2003.
[33] B. Ralph, “Modal Normalisation of Classical Proofs,” Ph.D. dissertation, University of Bath, 2019.
[34] A. A. Tubella and A. Guglielmi, “Subatomic Proof Systems: Splittable Systems,” ACM Transactions on Computational Logic (TOCL), vol. 19, no. 1, pp. 1–33, 2018.
[35] K. Brünnler, “Locality for Classical Logic,” Notre Dame Journal of Formal Logic, vol. 47, no. 4, pp. 557–580, 2006. [Online]. Available: http://www.iam.unibe.ch/~kai/Papers/LocalityClassical.pdf
[36] J.-Y. Girard, “Proof-nets: The Parallel Syntax for Proof-theory,” in Logic and Algebra, A. Ursini and P. Agliano, Eds. Marcel Dekker, New York, 1996.
[37] A. Guglielmi and L. Straßburger, “Non-commutativity and MELL in The Calculus of Structures,” in Computer Science Logic, CSL 2001, ser. LNCS, L. Fribourg, Ed., vol. 2142. Springer-Verlag, 2001, pp. 54–68.
[38] L. Straßburger, “Linear logic and Noncommutativity in the Calculus of Structures,” Ph.D. dissertation, Technische Universität Dresden, 2003.
[39] A. Hughes, “Unification Nets: Canonical Proof Net Quantifiers,” in Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, 2018, pp. 540–549.
[40] A. A. Tubella and L. Straßburger, “Introduction to Deep Inference,” 2019, lecture notes for ESSLLI’19. [Online]. Available: https://hal.inria.fr/hal-02390267
[41] K. Brünnler and A. F. Tiu, “A Local System for Classical Logic,” in International Conference on Logic for Programming Artificial Intelligence and Reasoning. Springer, 2001, pp. 347–361.
[42] C. Retoré, “Pomset Logic as a Calculus of Directed Cographs,” INRIA, Research Report RR-3714, 1999. [Online]. Available: https://hal.inria.fr/inria-00072953
[43] V. Dazos and L. Regnier, “The Structure of Multiplicatives,” Archive for Mathematical Logic, vol. 28, no. 3, pp. 181–203, 1989.
[44] C. Retoré, “Handsome Proof-nets: Perfect Matchings and Cographs,” Theoretical Computer Science, vol. 294, no. 3, pp. 473–488, 2003.
[45] C. Retoré, “Handsome Proof-nets: R&B-Graphs, Perfect Matchings and Series-parallel Graphs,” INRIA, Research Report RR-3652, 1999. [Online]. Available: https://hal.inria.fr/inria-00073020
[46] L. Straßburger, “Deep Inference and Expansion Trees for Second-order Multiplicative Linear Logic,” Mathematical Structures in Computer Science, vol. 29, pp. 1030–1060, 2019.
[47] B. Ralph and L. Straßburger, “Towards a Combinatorial Proof Theory,” in International Conference on Automated Reasoning with Analytic Tableaux and Related Methods. Springer, 2019, pp. 259–276.
[48] F. Lamarche, “Exploring the Gap Between Linear and Classical Logic,” Theory and Applications of Categories, vol. 18, no. 18, pp. 473–535, 2007.
APPENDIX

A. Proof of Theorem 29

**Proof of Theorem 29** Write $fv(A)$ for the set of variables which occur free in $A$.

Note that the instances of $w, c$ in $\Phi_2$ are deep, but inside sequent contexts. Initially, if an instance of $wk \vdash \Gamma, A$ is followed by a rule in which $A$ is not in the principal formula, it can be permuted downwards. Otherwise, the proof can be transformed using the following rewriting rules.

\[
\begin{align*}
wk & \vdash \Gamma, A \quad \vdash B, \Delta \quad \Rightarrow \quad wk \vdash \Gamma, A \wedge B, \Delta \\
wk & \vdash \Gamma, A \quad \vdash \Gamma, A \vee B \\
wk & \vdash \Gamma \quad \vdash \Gamma, A[x/t] \quad \Rightarrow \quad wk \vdash \Gamma, \exists x.A \\
wk & \vdash \Gamma, A \quad \vdash \Gamma, \forall x.A \\
wk & \vdash \Gamma, A \wedge A \quad \Rightarrow \quad \vdash \Gamma, A
\end{align*}
\]

Note that in the case of $\vee$, we use the deep rule $w$ which can be permuted under all the rules. By using these rewriting rules, we can eventually get a derivation with all the instances of $wk$ and $w$ at the bottom. Now observe that the instances of $\text{ctr}$ in $\Phi$ can be transformed using the following rule:

\[
\begin{align*}
\text{ctr} & \vdash \Gamma, A, A \quad \Rightarrow \quad c \vdash \Gamma, A \wedge A \\
\vdash \Gamma, A \quad \Rightarrow \quad c \vdash \Gamma, A \vee A \\
\vdash \Gamma, A \quad \Rightarrow \quad \vdash \Gamma, A
\end{align*}
\]

Knowing that $c$ can be permuted under all the rules of $\text{MLL}^X$, we eventually obtain a derivation:

\[
\begin{align*}
\text{MLL}^X & \quad \Phi_i \\
\vdash \Gamma' \\
\{wk, w, c, \equiv\} & \equiv \Phi_2 \\
\vdash \Gamma
\end{align*}
\]

Note that $\equiv$ is required here since the permutation of formulas is implicit in $\text{MLL}^X$.

By transforming each sequent of $\Phi_2'$ into its corresponding formula, and by considering the following rewriting rule:

\[
wk \vdash \Gamma, A \quad \Rightarrow \quad w \vdash \vee(\Gamma)
\]

we obtain a derivation

\[
\begin{align*}
\text{MLL}^X & \quad \Phi_1 \\
\vdash \Gamma' \\
\{w, c, w, \equiv\} & \equiv \Phi_2 \\
\vdash \Gamma
\end{align*}
\]

where $\Phi_1$ can be obtained from $\Phi_2'$ by applying the $\vee$ rule. $\square$

B. Rule permutation for the proof of Lemma 37

We construct a rewriting system based on rule permutation on derivations in $\{w, w_\forall, ac, c_\forall, m, m_\forall, m_3, \equiv\}$ that allows us to reach a derivation of the form

\[
\begin{align*}
A \quad \{m, m_\forall, m_3, \equiv\} & \equiv A' \\
\{ac, c_\forall\} & \equiv B' \\
\{w, w_\forall, \equiv\} & \equiv B
\end{align*}
\]

from any derivation. Intuitively, we want to move all the instances of $r \in \{w, w_\forall\}$ downwards and all the instances of $r' \in \{m, m_\forall, m_3\}$ upwards.

We first study the interactions between two rules. Certain cases are unsolved at this stage, and they are considered later when we study the interactions between two non-$\equiv$ rule instances separated by $\equiv$. Only non-trivial cases are presented here:

- $r_1/r_2$, where $r_1 \in \{w, w_\forall\}$ and $r_2 \in \{ac, c_\forall, m, m_\forall, m_3\}$:

\[
\begin{align*}
w & \vdash \alpha \vee \alpha \Rightarrow \alpha \\
ac & \vdash \alpha \\
w & \vdash \alpha \vee \alpha \\
m & \vdash \alpha \vee \alpha \Rightarrow \alpha \\
w & \vdash \alpha \vee \alpha \\
ac & \vdash \alpha \\
w & \vdash \alpha \vee \alpha \\
m & \vdash \alpha \vee \alpha \\
ac & \vdash \alpha \\
w & \vdash \alpha \vee \alpha \\
m & \vdash \alpha \vee \alpha \\
ac & \vdash \alpha
\end{align*}
\]

Note that $\equiv$ is required here since the permutation of formulas is implicit in $\text{MLL}^X$. 

14
where in the last case, $x$ is not free in $A$.

- $r_1/r_2$, where $r_1 \in \{ac, c_v\}$ and $r_2 \in \{m, m_v, m_3\}$:

\[
\frac{S[(\forall x.(\forall x.A) \lor (\forall x.B))]}{c_v} \quad \frac{S[(\forall x.(\forall x.A \lor (\forall x.B))]}{m_v} \quad \frac{S[(\forall x.(\forall x.A \lor B))]}{S[(\forall x.(A \lor B))]}
\]

- $\equiv / c_v$:

\[
\frac{\forall x.x.y.A}{c_v} \quad \frac{\forall y.y.y.A}{c_v} \quad \frac{\forall y.(\forall x.A) \lor B}{c_v}
\]

where in the last two cases, $x$ is not free in $B$.

- $w / \equiv$:

\[
\frac{w.A}{A \lor B} \quad \frac{w.A}{B \lor A} \quad \frac{w.(A \lor B) \lor C}{A \lor (B \lor C)}
\]

- $\equiv / m$:

\[
\frac{(\forall x.(\forall x.A) \lor B)}{c_v} \quad \frac{(\forall x.(\forall x.A) \lor B)}{(\forall x.A) \lor B} \quad \frac{(\forall x.(\forall x.A \lor B))}{(\forall x.(A \lor B))}
\]

where in the last four cases, $x$ is not free in $B$.

- $w_v / \equiv$:

In the following two cases, we assume $x \neq y$ (otherwise they are trivial).

\[
\frac{\forall y.A}{w_v} \quad \frac{\forall x.(\forall x.A) \lor (\forall y.A)}{w_v} \quad \frac{\forall y.A}{w_v} \quad \frac{\forall x.(\forall x.A) \lor (\forall y.A)}{w_v}
\]

where in the last two cases, the constraint on $x$ on the left-hand side implies that of the right-hand side.
\[
\begin{align*}
\forall y. \forall x. A \lor (\forall x. B) & \\equiv \forall x. (\forall y. A) \lor (\forall x. B) \\
\forall x. (\forall y. A) \lor (\forall x. B) & \\equiv (\forall x. A \lor (\forall x. B)) \\
(\forall x. A) \lor (\forall x. B) & \\equiv \forall x. (A \lor (\forall x. B)) \\
\forall x. (A \lor (\forall x. B)) & \\equiv \forall x. (A \lor (\forall y. A)) \\
\forall x. (A \lor (\forall y. A)) & \\equiv \forall x. (A \lor (\forall x. A)) \\
\forall x. (A \lor (\forall x. A)) & \\equiv (\forall x. A) \lor (\forall x. A)
\end{align*}
\]

- \(\equiv /m_3\): similar to \(\equiv /m_v\).

Interactions between two non-\(\equiv\) rules with the presence of \(\equiv\) in between:

- \(\forall / \equiv / r\) where \(r \in \{m, m_v, m_3\}\): First permute \(\forall v\) under \(\equiv\) and then permute \(\forall v\) under \(r\).
- \(ac / \equiv / r\) where \(r \in \{m, m_v, m_3\}\): First permute \(ac\) under \(\equiv\) and then permute \(ac\) under \(r\).
- \(w / \equiv / \forall v:\)

\[
\begin{align*}
\forall x. \forall y. A & \\equiv \forall x. (\forall y. A) \lor (\forall x. B) \\
\forall x. (\forall y. A) \lor (\forall x. B) & \\equiv (\forall x. A) \lor (\forall x. B) \\
(\forall x. A) \lor (\forall x. B) & \\equiv \forall x. (A \lor (\forall x. B)) \\
\forall x. (A \lor (\forall x. B)) & \\equiv \forall x. (A \lor (\forall y. A)) \\
\forall x. (A \lor (\forall y. A)) & \\equiv \forall x. (A \lor (\forall x. A)) \\
\forall x. (A \lor (\forall x. A)) & \\equiv (\forall x. A) \lor (\forall x. A)
\end{align*}
\]

where in all four cases, \(x\) is not free in \(B\).

- \(w / \equiv / ac:\)

\[
\begin{align*}
\forall x. A & \\equiv (\forall x. A) \lor (\forall x. B) \\
(\forall x. A) \lor (\forall x. B) & \\equiv \forall x. (A \lor (\forall x. B)) \\
\forall x. (A \lor (\forall x. B)) & \\equiv (\forall x. A) \lor (\forall x. A)
\end{align*}
\]

\[
\begin{align*}
\forall x. B & \\equiv (\forall x. B) \lor (\forall x. A) \\
(\forall x. B) \lor (\forall x. A) & \\equiv \forall x. (B \lor (\forall x. A)) \\
\forall x. (B \lor (\forall x. A)) & \\equiv (\forall x. A) \lor (\forall x. B)
\end{align*}
\]

\[
\begin{align*}
\forall x. (A \lor (\forall x. A)) & \\equiv \forall x. (A \lor (\forall x. B)) \\
\forall x. (A \lor (\forall x. B)) & \\equiv (\forall x. A) \lor (\forall x. B)
\end{align*}
\]

\[
\begin{align*}
\exists x. B & \\equiv (\exists x. B) \lor (\exists x. A) \\
(\exists x. B) \lor (\exists x. A) & \\equiv \exists x. (B \lor (\exists x. A)) \\
\exists x. (B \lor (\exists x. A)) & \\equiv (\exists x. A) \lor (\exists x. B)
\end{align*}
\]

C. Proof of Lemma 43

Proof of Lemma 43 By [47, Proposition 7.5], there is a derivation \(\{ac, m, m_v, m_3, \equiv\}\) \(H^0\). We plan to show that \(\Psi\) can be lifted to \(\{ac, c, m, m_v, m_3, \equiv\}\). However, observe that not every formula occurring in \(\Psi\) is a propositional encoding. There are two reasons for this: (i) we might have \(P \equiv^c \equiv^m Q\) where \(P\) is a propositional encoding but \(Q\) is not, and (ii) the rule \(ac\) can duplicate an atom \(x \in \text{VAR}\). Let us write \(ac_x\) for such instances. The problem with (i) is that we could have the following situation

\[
\begin{align*}
S\{(x \land (E \land C)) \lor (x \land (F \land D))\} & \equiv^c S\{(x \land E) \land C \lor (x \land F) \lor D\} \\
S\{(x \land E) \lor (x \land F) \lor D\} & \equiv^m S\{(x \land E) \land C \lor (x \land F) \lor D\}
\end{align*}
\]

(7)
where \( x \) occurs in \( C \lor D \). Then premise and conclusion are both propositional encodings, but the whole derivation cannot be lifted. However, since we demand that the mapping is a fibration (and therefore a homomorphism) on the binding graphs, there must be another instance of \( m \) further below in the derivation:

\[
\begin{align*}
&\text{m} \\
&S'(\{x \land E\} \lor (x \land F)) \\
&\equiv S'(x \lor x) \land (E \lor F)).
\end{align*}
\]

(8)

We can permute both instances via the following more general scheme (see \([23, 43]\) for a general discussion on permutations of the m-rule):

\[
\begin{align*}
&\text{m} \\
&S((G \land E \lor C) \lor (G \land F \lor D)) \\
&\equiv S((G \lor H) \land (E \lor F) \land (C \lor D)).
\end{align*}
\]

(9)

We omitted some instances of \( \equiv^0 \) and some parentheses. We now call instances of \( m \) as in (7) \text{ illegal}, and we can transform \( \Psi \) through \( m \)-permutations into a derivation that does not contain any illegal \( m \)-instances. To address (ii), we also apply a permutation argument, permuting all instances of \( \text{ac}_x \) up until they either reach the top of the derivation or an instance of \( m \) which separates the two atoms in the premise. More precisely, we consider the following inference rule

\[
\text{ac}_x \equiv S_0(\{x \lor S_2(x)\}) \\
\]

\[
\text{ac}_x \equiv S_1(\{x \lor S_2(x)\}) \\
\]

where \( S_1(\cdot) \equiv (\cdot) \lor E \) and \( S_2(\cdot) \equiv (\cdot) \lor F \) and \( S(\cdot) \equiv S_0(\{\cdot \lor E \lor F\}) \) for some formulas \( E \) and \( F \), where \( E \lor F \) or both might be empty. The rule \( \text{ac}^\equiv_x \) permutes over \( \equiv, \text{ac} \), and other instances of \( \text{ac}^\equiv_x \), and over instances of \( m \) if they occur inside \( S_0 \) or \( S_1 \) or \( S_2 \). The only situation in which \( \text{ac}^\equiv_x \) cannot be permuted up is the following:

\[
\begin{align*}
&\text{m} \\
&S((R_1(\{x \lor C\} \lor (R_2(\{x \lor D) \land D)) \land (C \lor D)) \\
&\equiv S(R(\{x \lor C \lor (C \lor D)) \land (C \lor D)).
\end{align*}
\]

(11)

We can therefore assume that all instances of \( \text{ac}_x \), that contract an atom \( x \in \text{VAR} \) are either at the top of \( \Psi \) or below a \( m \)-instance as in (11). We now lift \( \Psi \) to \( \{\text{ac}, \text{cv}, m, m_v, m_3, \equiv\} \), proceed by induction on the height of \( \Psi \), beginning at the top, making a case analysis on the topmost rule that is not a \( \equiv \).

- \( \text{ac}_x \): We know that the premise of (10) is a propositional encoding. Hence, \( S_1(\cdot) \equiv (\cdot) \lor E^o \) and \( S_2(\cdot) \equiv (\cdot) \lor F^o \) and both \( x \) are universal, and \( E^o \lor F^o \) contains all occurrences of \( x \) bound by that universal. We have the following subcases:
  - \( E \) and \( F \) are both non-empty: We have

\[
\text{ac}_x \equiv S^o((x \lor E^o) \lor (x \lor F^o)) \\
\equiv S^o(x \lor (E^o \lor F^o))
\]

which can be lifted to

\[
\text{m} \equiv S((\forall x. E) \lor (\forall x. F)) \\
\equiv S(\forall x.(E \lor F))
\]

where \( S^o(\cdot), E^o, F^o \) are the propositional encodings of \( S(\cdot), E, F \), respectively.

- \( E^o \) is empty and \( F^o \) is non-empty: We have

\[
\text{ac}_x \equiv S^o((x \lor (E^o \lor F^o)) \\
\equiv S^o(x \lor F^o)
\]

which can be lifted to

\[
\text{cv} \equiv S((\forall x.\forall x.E) \lor (\forall x.\forall x.F)) \\
\equiv S(\forall x.\forall x(F))
\]

- \( E^o \) is non-empty and \( F^o \) is empty: This is similar to the previous case.

- \( E^o \) and \( F^o \) are both empty: This is impossible as the premise would not be a propositional encoding.

- \( \text{ac} \) (contracting an ordinary atom): This can trivially be lifted.

- \( m \): There are several cases to consider.
  - If none of the four principal formulas in the premise is \( x \lor x \lor F \) for some formula \( F \) and \( x \in \text{VAR} \), then this instance of \( m \) can trivially be lifted, and we can proceed by induction hypothesis.
  - If exactly one of the four principal formulas in the premise is \( x \) for some \( x \in \text{VAR} \), then this \( x \) is the encoding of an existential in the premise and of an universal in the conclusion. This is impossible, as \( \varphi \) has to preserve existentials.
  - If two of the four principal formulas in the premise are \( x \) for some \( x \in \text{VAR} \), then we are in the following special case of (11):

\[
\begin{align*}
&\text{m} \\
&S((x \lor C) \lor (x \lor D)) \\
&\equiv S((x \lor D) \land (C \lor D))
\end{align*}
\]

which can be lifted immediately to

\[
\begin{align*}
&\text{m}_3 \\
&S((3 \lor C) \lor (3 \lor D)) \\
&\equiv S((3 \lor C) \lor (3 \lor D))
\end{align*}
\]

- We have a situation (11) where \( R_1(\{x \lor E) \equiv x \lor E \lor F ) \) for some \( E \) and \( R_2(\{x \lor D) \equiv x \lor E \lor F \) for some \( F \) with \( R(\{x \lor E \lor F ) \equiv x \lor E \lor F \) (Otherwise, the application of \( \text{ac}^\equiv_x \) would not be correct.) That means, we have:

\[
\begin{align*}
&\text{m} \\
&S((((x \lor E) \lor C) \lor ((x \lor F) \lor D)) \\
&\equiv S(((x \lor E) \lor (x \lor F)) \land (C \lor D))
\end{align*}
\]

which can be lifted to

\[
\begin{align*}
&\text{m}_3 \\
&S(((\forall x.E) \lor (\forall x.F)) \lor (C \lor D)) \\
&\equiv S(\forall x.(E \lor F)) \lor (C \lor D))
\end{align*}
\]

- In all other cases (e.g. exactly one of the principal formulas is of shape \( x \lor F \) (and none is \( x \)), we can trivially lift the \( m \)-instance, as the quantifier structure is not affected.

Thus \( \Psi \) can be lifted to \( \{\text{ac}, \text{cv}, m, m_v, m_3, \equiv\} \parallel \Phi \).