AN ANALYTIC PROPOSITIONAL PROOF SYSTEM ON GRAPHS

MATTEO ACCLAVIO, ROSS HORNE, AND LUTZ STRASSBURGER

[a] Dipartimento di Matematica e Fisica, Università Roma Tre, Roma, Italy
 URL: matteoacclavio.com/Math.html

[b] Computer Science, University of Luxembourg, Esch-sur-Alzette, Luxembourg,
 e-mail address: ross.horne@uni.lu

[c] Inria, Equipe Partout, Ecole Polytechnique, LIX UMR 7161, France
 URL: www.lix.polytechnique.fr/Labo/Lutz.Strassburger/

ABSTRACT. In this paper we present a proof system that operates on graphs instead of formulas. Starting from the well-known relationship between formulas and cographs, we drop the cograph-conditions and look at arbitrary (undirected) graphs. This means that we lose the tree structure of the formulas corresponding to the cographs, and we can no longer use standard proof theoretical methods that depend on that tree structure. In order to overcome this difficulty, we use a modular decomposition of graphs and some techniques from deep inference where inference rules do not rely on the main connective of a formula. For our proof system we show the admissibility of cut and a generalisation of the splitting property. Finally, we show that our system is a conservative extension of multiplicative linear logic with mix, and we argue that our graphs form a notion of generalised connective.

CONTENTS

1. Introduction

The notion of formula is central to all applications of logic and proof theory in computer science, ranging from the formal verification of software, where a formula describes a property that the program should satisfy, to logic programming, where a formula represents a program [MNPS91, KY93], and functional programming, where a formula represents a type [How80]. Proof theoretical methods are also employed in concurrency theory, where a formula can represent a process whose behaviours may be extracted from a proof of the formula [Mil93, Bru02, FRS01, DSC16, OP17, NOP17, HT19, Hor19, Hor20]. This *formulas-as-processes* paradigm is not as well-investigated as the *formulas-as-properties*, *formulas-as-programs* and *formulas-as-types* paradigms mentioned before. In our opinion, a reason for this is that the notion of formula reaches its limitations when it comes to describing processes as they are studied in concurrency theory.

For example, Guglielmi's BV [Gug07] and Retoré's *pomset logic* [Ret97] are proof systems which extend linear logic with a notion of sequential composition and can model series-parallel orders.[1,2] However, series-parallel orders cannot express some ubiquitous patterns of *causal dependencies* such as the logical time constraints on producer-consumer queues [LW00], which are within the scope of pomsets [Pra86], event structures [NPW81], and Petri nets [Pet77]. The essence of this problem is already visible when we consider *symmetric dependencies*, such as separation, which happens to be the dual concept to concurrency in the *formulas-as-processes* paradigm.

Let us use some simple examples to explain the problem. Suppose we are in a situation where two processes A and B can communicate with each other, written as $A \bindnasrepma B$, or are separated from each other, written as $A \otimes B$, such that no communication is possible. Now assume we have four atomic processes a, b, c, and d, from which we form the two processes $P = (a \otimes b) \bindnasrepma (c \otimes d)$ and $Q = (a \bindnasrepma c) \otimes (b \bindnasrepma d)$. Both are perfectly fine formulas of multiplicative linear logic (MLL) [Gir87a]. In P, we have that a is separated from b but can communicate with c and d. Similarly, d can communicate with a and b but is separated from c, and so on. On the other hand, in Q, a can only communicate with c and is separated from the other two, and d can only communicate with b, and is separated from the other two. We can visualise this situation via graphs where a, b, c, and d are the vertices, and we draw an edge between two vertices if they are separated, and no edge if they can communicate. Then P and Q correspond to the two graphs shown below.

$$P = (a \otimes b) \bindnasrepma (c \otimes d) \qquad Q = (a \bindnasrepma c) \otimes (b \bindnasrepma d) \tag{1.1}$$

It should also be possible to describe a situation where a is separated from b, and b is separated from c, and c is separated from d, but a can communicate with c and d, and b can communicate with d, as indicated by the graph below.

$$\tag{1.2}$$

---

[1] In his PhD-thesis [Ret93], Retoré considers all partial ordered multisets, but in later versions [Ret21] only series-parallel orders are considered to maintain the correspondence to formulas.

[2] It has long been believed that BV and pomset logic are the same, but recently it has been shown that this is not the case [NS22a, NS22b].

An example of this behaviour could arise in the setting of concurrent processes, where four processes a, b, c, and d satisfy information flow constraints such that c and a can communicate; a and d can communicate; and d and b can communicate, and no further communications are possible. This models an intransitive information flow, since we have two processes, a and d, which can communicate with each other, and respectively with c and b; yet the same processes must also ensure that no information flows between c and b. However, the graph (1.2) cannot be described by a formula in the way illustrated for the two graphs in (1.1).

This means that the tools of proof theory, which have been developed over the course of the last century and which were very successful for the *formulas-as-properties*, *formulas-as-programs*, and *formulas-as-types* paradigms, cannot be used for the *formulas-as-processes* paradigm unless we forbid situations as in (1.2) above. This seems to be a very strong and unnatural restriction (that is, it is an *a posteri* restriction imposed by the use of formulas, with no *a priory* justification stemming from process modelling problems). The purpose of this paper is to propose a way to change this unsatisfactory situation.

We will present a proof system, called GS (for *graphical proof system*), whose objects of reason are not formulas but graphs, giving the example in (1.2) the same status as the examples in (1.1). In a less informal way, one could say that standard proof systems work on cographs (which are the class of graphs that correspond to formulas as in (1.1) [Duf65]), and our proof systems works on arbitrary graphs. In order for this to make sense, our proof system should obey the following basic properties:

(1) *Consistency*: There are graphs that are not provable. In particular, if only a finite number of graphs is provable (or not provable) then the proof system would not be interesting.

(2) *Transitivity*: The proof system should come with an implication that is transitive, i.e., if we can prove that A implies B and that B implies C, then we should also be able to prove that A implies C.

(3) *Analyticity*: As we no longer have formulas, we cannot ask that every formula that occurs in a proof is a subformula of its conclusion. However, we can investigate a graph theoretical version of this idea, and we can ask that in a proof search situation, there is always only a finite number of ways to apply an inference rule.

(4) *Minimality*: We want to make as few assumptions as possible, so that the theory we develop is as general as possible.

Properties 1-3 are standard for any proof system, and they are usually proved using cut elimination. In that respect our paper is no different. We introduce a notion of cut and show its admissibility for GS. Then Properties 1-3 are immediate consequences.

Property 4 is of a more subjective nature. In our case, we only make the following two basic assumptions:

(i) For any graph A, we should be able to prove that A implies A. This assumption is almost impossible to argue against, so can be expected for any logic.

(ii) If a graph A is provable, then the graph $G = C[A]$ is also provable, provided that $C[\cdot]$ is a provable context.[3] This can be compared to the *necessitation rule* of modal logic, which says that if A is provable then so is $\Box A$, except that in our case the $\Box$ is replaced by the provable graph context $C[\cdot]$.

---

[3] Formally, the notation $G = C[A]$ means that A is a module of G, and $C[\cdot]$ is the graph obtained from G by removing all vertices belonging to A. We give the formal definition in Section 3.

All other properties of the system GS follow from the need to obtain admissibility of cut. This means that this paper does not present some random system, but follows the underlying principles of proof theory. For a more detailed philosophical presentation of these principles, we refer the reader to Appendix B.

We also target the desirable property of *conservativity*. This means that there should be a well-known logic L (based on formulas) such that the restriction of our proof system to those graphs that correspond to formulas proves exactly the theorems of L. This cannot be an assumption used to design a logical system, since it would create circularity (to specify a logic we need a logic); conservativity is more so a cultural sanity condition to check that we have not invented an esoteric logic. In our case, conservativity will follow from cut admissibility, and the logic L is multiplicative linear logic with mix (MLL°) [Gir87a, Bel97, FR94].

Let us now summarise how this paper is organised: In Section 2, we give preliminaries on cographs, which form the class of graphs that correspond to formulas as in (1.1). Then, in Section 3 we give some preliminaries on modules and prime graphs, which are needed for our move away from cographs, so that in Section 4, we can present our proof system, which uses the notation of open deduction [GGP10] and follows the principles of deep inference [GS01, BT01, Gug07]. To our knowledge, this is the first proof system that is not tied to formulas/cographs but handles arbitrary (undirected) graphs instead. In Section 5 we show some properties of our system, and Sections 6 and 7 are dedicated to cut elimination, which is the basis for showing properties (1), (2), and (3), mentioned above. We also explain the technology we must develop in order to be able to prove cut elimination for our proof system. The interesting point is that, not only do we go beyond methods developed for the sequent calculus, but we also go beyond methods developed for deep inference on formulas. In particular, we require entirely new statements of the tools called *splitting* and *context reduction*, and furthermore their proofs are inter-dependent, whereas normally context reduction follows from splitting [GS11, ATG18].

Then, in Section 8, we not only show that our system is a conservative extension of MLL°, we also show a form of analyticity for our system. Finally, in Section 9, we show how our work is related to the work on generalised connectives introduced in [Gir87b, DR89]. We end this paper with a discussion of related work in Section 10 and a conclusion in Section 11.

Compared to the conference version [AHS20] of this paper, there are the following three major additions:

- We give detailed proofs of the *Splitting Lemma* and the *Context Reduction Lemma* (in Section 6 and Appendix A), which are crucial for the cut elimination proof. In fact, we also completely reorganised the proofs with respect to the technical appendix of [AHS20][4]. For proving these lemmas, we could not rely on the general method that has been proposed by Aler Tubella in her PhD [AT17].
- We present a notion of analyticity for proof systems on graphs and show that our system is analytic in that respect (in Section 8).
- We show that general graphs with n vertices can be seen as generalised n-ary connectives (in Section 9), and we compare this notion with the existing notion of generalised (multiplicative) connective [Gir87b, DR89, Mai19, AM20].
- We include in Appendix B a more detailed discussion of how our proof system can be considered satisfactory with respect to the Properties (1)–(4) mentioned above.

---

[4]That appendix is available at `https://hal.inria.fr/hal-02560105`.

Finally, let us argue that logics are not designed but discovered. They typically follow logical principles where design parameters are limited. For example, we will see that we do not get to chose whether or not the following implications hold:

$$\nvdash \quad
\begin{matrix} b & d \\ | & | \\ a & c \end{matrix}
\;\multimap\;
\begin{matrix} b & d \\ | & | \\ a & c \end{matrix}
\qquad\qquad
\vdash \quad
\begin{matrix} b & d \\ | & | \\ a & c \end{matrix}
\;\multimap\;
\begin{matrix} b & d \\ | & | \\ a & c \end{matrix}
\tag{1.3}$$

There is no pre-existing semantics or proof system we can refer to at this point. Nonetheless, from the above discussed principles we can argue, that in a logic on graphs, the former implication in (1.3) cannot hold while the latter must hold. Over the course of this paper, we explore the design of proof systems on graphs based on logical principles, which enables us to confidently state such facts.

2. From Formulas to Graphs

In this preliminary section we recall the basic textbook definitions for graphs and formulas, and their correspondence via cographs.

Definition 2.1. A *(simple, undirected) graph* G is a pair $\langle V_G, E_G \rangle$ where V_G is a set of vertices and E_G is a set of two-element subsets of V_G. We omit the index G when it is clear from the context. For $v, w \in V_G$ we write vw as an abbreviation for $\{v, w\}$. A graph G is *finite* if its vertex set V_G is finite. Let L be a set and G be a graph. We say that G is *L-labelled* (or just *labelled* if L is clear from the context) if every vertex in V_G is associated with an element of L, called its *label*. We write $\ell_G(v)$ to denote the label of the vertex v in G. A graph G' is a *subgraph* of a graph G, denoted as $G' \subseteq G$ iff $V_{G'} \subseteq V_G$ and $E_{G'} \subseteq E_G$. We say that G' is an *induced subgraph* of G if G' is a subgraph of G and for all $v, w \in V_{G'}$, if $vw \in E_G$ then $vw \in E_{G'}$. For a graph G we write $|V_G|$ for its number of vertices and $|E_G|$ for its number of edges.

In the following, we will just say *graph* to mean a finite, undirected, labelled graph, where the labels come from the set $\mathcal{A}$ of atoms which is the (disjoint) union of a countable set of propositional variables $\mathcal{V} = \{a, b, c, \ldots\}$ and their duals $\mathcal{V}^\perp = \{a^\perp, b^\perp, c^\perp, \ldots\}$.

Since we are mainly interested in how vertices are labelled, but not so much in the identity of the underlying vertex, we heavily rely on the notion of graph isomorphism.

Definition 2.2. Two graphs G and G' are *isomorphic* if there exists a bijection $f \colon V_G \to V_{G'}$ such that for all $v, u \in V_G$ we have $vu \in E_G$ iff $f(v)f(u) \in E_{G'}$ and $\ell_G(v) = \ell_{G'}(f(v))$. We denote this as $G \simeq_f G'$, or simply as $G \simeq G'$ if f is clear from the context or not relevant.

In the following, we will, in diagrams, forget the identity of the underlying vertices, showing only the label, as in the examples in the introduction.

In the rest of this section we recall the characterisation of those graphs that correspond to formulas. For simplicity, we restrict ourselves to only two connectives, and for reasons that will become clear later, we use the $\mathbin{\rotatebox[origin=c]{180}{\&}}$ (*par*) and $\otimes$ (*tensor*) of linear logic [Gir87a]. More precisely, *formulas* are generated by the grammar

$$\phi, \psi ::= \circ \mid a \mid a^\perp \mid \phi \mathbin{\rotatebox[origin=c]{180}{\&}} \psi \mid \phi \otimes \psi \tag{2.1}$$

where $\circ$ is the *unit*, and a can stand for any propositional variable in $\mathcal{V}$. As usual, we can define the negation of formulas inductively by letting $a^{\perp\perp} = a$ for all $a \in \mathcal{V}$, and by using

the De Morgan duality between $\mathbin{⅋}$ and $\otimes$: $(\phi \mathbin{⅋} \psi)^\perp = \phi^\perp \otimes \psi^\perp$ and $(\phi \otimes \psi)^\perp = \phi^\perp \mathbin{⅋} \psi^\perp$; the unit is self-dual: $\circ^\perp = \circ$.

On formulas we define the following structural equivalence relation:

$$
\begin{array}{cc}
\phi \mathbin{⅋} (\psi \mathbin{⅋} \xi) \equiv (\phi \mathbin{⅋} \psi) \mathbin{⅋} \xi & \phi \otimes (\psi \otimes \xi) \equiv (\phi \otimes \psi) \otimes \xi \\
\phi \mathbin{⅋} \psi \equiv \psi \mathbin{⅋} \phi & \phi \otimes \psi \equiv \psi \otimes \phi \\
\phi \mathbin{⅋} \circ \equiv \phi & \phi \otimes \circ \equiv \phi
\end{array}
\tag{2.2}
$$

In order to translate formulas to graphs, we define the following two operations on graphs:

Definition 2.3. Let $G = \langle V_G, E_G \rangle$ and $H = \langle V_H, E_H \rangle$ be graphs. We define the **par** of G and H to be their disjoint union and the **tensor** to be their join, i.e.:

$$
\begin{aligned}
G \mathbin{⅋} H &= \langle V_G \uplus V_H, E_G \uplus E_H \rangle \\
G \otimes H &= \langle V_G \uplus V_H, E_G \uplus E_H \uplus \{vw \mid v \in V_G, w \in V_H\} \rangle
\end{aligned}
$$

These operations can be visualised as follows:

$$\tag{2.3}$$

For a formula ϕ, we can now define its associated graph $\llbracket \phi \rrbracket$ inductively as follows: $\llbracket \circ \rrbracket = \varnothing$ the empty graph; $\llbracket a \rrbracket = a$ a single-vertex graph whose vertex is labelled by a (by a slight abuse of notation, we denote that graph also by a); similarly $\llbracket a^\perp \rrbracket = a^\perp$; finally we define $\llbracket \phi \mathbin{⅋} \psi \rrbracket = \llbracket \phi \rrbracket \mathbin{⅋} \llbracket \psi \rrbracket$ and $\llbracket \phi \otimes \psi \rrbracket = \llbracket \phi \rrbracket \otimes \llbracket \psi \rrbracket$.

Theorem 2.4. *For any two formulas, $\phi \equiv \psi$ iff $\llbracket \phi \rrbracket \simeq \llbracket \psi \rrbracket$.*

Proof. By a straightforward induction. $\qquad\square$

Definition 2.5. A graph is $\mathsf{P_4}$-*free* iff it does not have an induced subgraph of the shape

$$\tag{2.4}$$

The following result is classical and its proof can be found, e.g., in [Möh89] or [Gug07].

Theorem 2.6 ([Duf65]). *Let G be a graph. Then there is a formula ϕ with $\llbracket \phi \rrbracket \simeq G$ iff G is $\mathsf{P_4}$-free.*

The graphs characterised by Theorem 2.6 are called **cographs**, because they are the smallest class of graphs containing all single-vertex graphs and being closed under complement and disjoint union.

Because of Theorem 2.6, one can think of standard proof systems as *cograph proof systems*. Since in this paper we want to move from cographs to general graphs, we need to investigate how much of the tree structure of formulas (which makes cographs so interesting for proof theory [Ret03, Hug06, Str17]) can be recovered for general graphs.
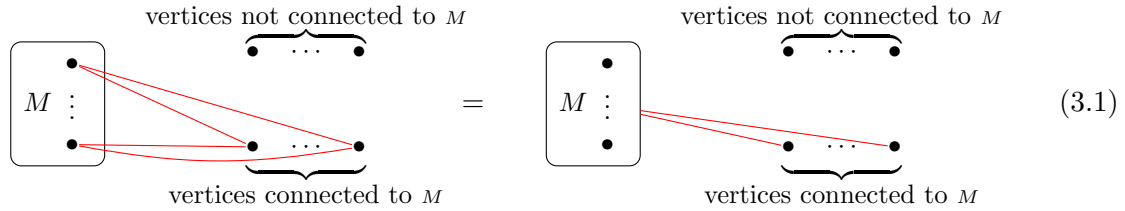
3. Modules and Prime Graphs

One of the consequences of the tree structure of formulas is the notion of *subformula*, which is induced by the notion of subtree. This is lost if we move from cographs to general graphs. However, in a cograph G, a *module* of G corresponds to a subformula of the formula ϕ given by Theorem 2.6. The notion of module also exists for general graphs, and in our proof systems on graphs, modules will play a similar role as subformulas play in ordinary proof systems on formulas.
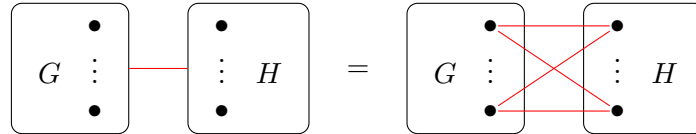
For this reason, we recall here some standard results on graph modules and the modular decomposition of graphs [MR84, EHR99, CD08, HP10].

Definition 3.1. Let G be a graph. A **module** of G is an induced subgraph $M = \langle V_M, E_M \rangle$ of G such that for all $v \in V_G \setminus V_M$ and all $x, y \in V_M$ we have $vx \in E_G$ iff $vy \in E_G$.

Notation 3.2. Let M be a module of a graph G. Since each vertex in V_M has the same relation with all vertices outside M, that is, with all vertices in $V_G \setminus V_M$, we introduce the following notation when drawing graphs

(3.1)

which allows us to reduce the number of drawn edges and to increase readability. Using this notation the definition of the tensor operator in (2.3) can be written as folllows:

Notation 3.3. Let M be a module of a graph G, and let C be the graph obtained from G by removing all vertices in M (including incident edges). If $R \subseteq V_C$ is the set of vertices that are in G connected to a vertex in V_M (and hence to all vertices in M by modularity), then we write $G = C[M]_R$ and we say that $C[\cdot]_R$ is the **context** of M in G.

Alternatively, we can consider a context $C[\cdot]_R$ as a graph with a distinguished vertex $[\cdot]$ such that R is the set of neighbours of $[\cdot]$. In this case $C[M]_R$ can be defined as the graph obtained from $C[\cdot]R$ by substituting the vertex $[\cdot]$ with the graph M and adding all the edges from each vertex in V_M to each vertex in R.

Example 3.4. Consider the graph context on the left below with the hole denoted by an empty ⬜. If we substitute the two-vertex graph $a \,⅋\, a^\perp$ for the hole, then we obtain the graph in the middle below:

Finally, if we replace the hole with the empty graph we obtain the graph on the right above.

Lemma 3.5. *Let G be a graph and M, N be modules of G. Then*
(1) $M \cap N$ *is a module of G;*

(2) *if $M \cap N \neq \varnothing$, then $M \cup N$ is a module of G; and*
(3) *if $N \nsubseteq M$ then $M \setminus N$ is a module of G.*

Proof. The first statement follows immediately from the definition. For the second one, let $L = M \cap N \neq \varnothing$, and let $v \in G \setminus (V_M \cup V_N)$ and $x, y \in V_M \cup V_N$. If x, y are both in M or both in N, then we have immediately $vx \in E_G$ iff $vy \in E_G$. So, let $x \in V_M$ and $y \in V_N$, and let $z \in L$. We have $vx \in E_G$ iff $vz \in E_G$ iff $vy \in E_G$. Finally, for the last statement, let $x, y \in V_M \setminus V_N$ and let $v \in V_G \setminus (V_M \setminus V_N)$. If $v \notin V_M$, we immediately have $vx \in E_G$ iff $vy \in E_G$. So, let $v \in V_M$, and therefore $v \in V_M \cap V_N$. Let $z \in V_N \setminus V_M$. Then $vx \in E_G$ iff $zx \in E_G$ iff $zy \in E_G$ iff $vy \in E_G$. □

Definition 3.6. Let G be a graph. A module M in G is **maximal** if $M \neq G$ and for all modules $M' \neq G$ of G we have that $M \subseteq M'$ implies $M = M'$.

Definition 3.7. A module M of a graph G is **trivial** iff either $V_M = \varnothing$ or V_M is a singleton or $V_M = V_G$. A graph G is **prime** iff $|V_G| \geq 2$ and all modules of G are trivial.

Definition 3.8. Let G be a graph with n vertices $V_G = \{v_1, \ldots, v_n\}$ and let $H_1, \ldots, H_n$ be n graphs. We define the **composition of $H_1, \ldots, H_n$ via G**, denoted as $G(\!|H_1, \ldots, H_n|\!)$, by replacing each vertex v_i of G by the graph H_i; and there is an edge between two vertices x and y if either x and y are in the same H_i and $xy \in E_{H_i}$ or $x \in V_{H_i}$ and $y \in V_{H_j}$ for $i \neq j$ and $v_i v_j \in E_G$. Formally, $G(\!|H_1, \ldots, H_n|\!) = \langle V^*, E^* \rangle$ with

$$V^* = \biguplus_{1 \leq i \leq n} V_{H_i} \quad \text{and} \quad E^* = \left(\biguplus_{1 \leq i \leq n} E_{H_i} \right) \uplus \left\{ xy \mid x \in V_{H_i}, y \in V_{H_j}, v_i v_j \in E_G \right\}$$

This concept allows us to decompose graphs into prime graphs (via Lemma 3.9 below) and recover a tree structure for an arbitrary graph.[5] The two operations $⅋$ and $\otimes$, defined in Definition 2.3 are then represented by the following two prime graphs:

$$⅋: \quad \bullet \quad \bullet \quad \text{and} \quad \otimes: \quad \bullet\!\!-\!\!\!-\!\!\bullet \tag{3.2}$$

We have $⅋(\!|G, H|\!) = G ⅋ H$ and $\otimes(\!|G, H|\!) = G \otimes H$.

Lemma 3.9 ([Gal67]). *For every non-empty graph G we have exactly one of the following four cases:*

(i) *G is a singleton graph.*
(ii) *$G = A ⅋ B$ for some A, B with $A \neq \varnothing \neq B$.*
(iii) *$G = A \otimes B$ for some A, B with $A \neq \varnothing \neq B$.*
(iv) *$G = P(\!|A_1, \ldots, A_n|\!)$ for some prime graph P with $n = |V_P| \geq 4$ and $A_i \neq \varnothing$ for every $i \in \{1, \ldots, n\}$.*

Proof. Let G be given. If $|G| = 1$, we are in case (i). Now assume $|G| > 1$, and let $M_1, \ldots, M_n$ be the maximal modules of G. Note that $n \geq 2$. Now we have two cases:

• For all $i, j \in \{1, \ldots, n\}$ with $i \neq j$ we have $M_i \cap M_j = \varnothing$. Since every vertex of G forms a module, every vertex must be part of a maximal module. Hence $V_G = V_{M_1} \cup \cdots \cup V_{M_n}$. Therefore there is a graph P such that $G = P(\!|M_1, \ldots, M_n|\!)$. Since all M_i are maximal in G, we can conclude that P is prime. If $|V_P| \geq 4$ we are in case (iv). If $|V_P| < 4$ we are either in case (ii) or (iii), as the two graphs in (3.2) are only two prime graphs with $|V_P| = 2$, and there are no prime graphs with $|V_P| = 3$.
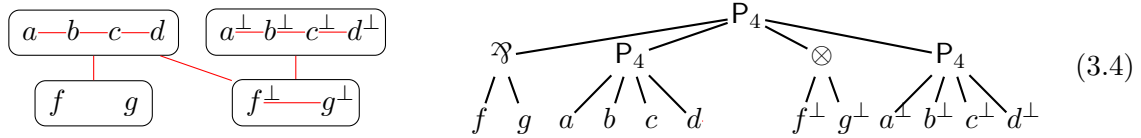
---

[5] This also allows us to consider prime graphs as generalised non-decomposable n-ary connectives. We will come back to this point in Section 9.

- We have some $i \neq j$ with $M_i \cap M_j \neq \varnothing$. Let $L = M_i \cap M_j$ and $N = M_i \setminus M_j$ and $K = M_j \setminus M_i$. By Lemma 3.5, L, N, K, and $M_i \cup M_j$ are all modules of G. Since M_i and M_j are maximal, it follows that $G = M_i \cup M_j$, and therefore $G = N \otimes L \otimes K$ or $G = N \mathbin{⅋} L \mathbin{⅋} K$. □

Example 3.10. Consider the following graph

$$a\text{—}b\text{—}c\text{—}d \quad a^\perp\text{—}b^\perp\text{—}c^\perp\text{—}d^\perp \tag{3.3}$$

Its representation using the modular notation introduced in (3.1) is shown on the left below and its modular decomposition tree is shown on the right below:

$$\tag{3.4}$$

The modular decomposition tree can also be written in "formula style" as

$$\mathsf{P}_4(\!(f \mathbin{⅋} g, \mathsf{P}_4(\!(a, b, c, d)\!), f^\perp \otimes g^\perp, \mathsf{P}_4(\!(a^\perp, b^\perp, c^\perp, d^\perp)\!))\!) \tag{3.5}$$

In the rest of the paper we will use all four representations in (3.3) and (3.4) and (3.5) interchangeably.

4. The Proof System

To define a proof system on graphs, we need a notion of implication on graphs. To do so, we first introduce a notion of negation on graphs.

Definition 4.1. For a graph $G = \langle V_G, E_G \rangle$, we define its **dual** $G^\perp = \langle V_G, E_{G^\perp} \rangle$ to have the same set of vertices, and an edge $vw \in E_{G^\perp}$ iff $vw \notin E_G$ (and $v \neq w$). The label of a vertex v in $G^\perp$ is the dual of the label of that vertex in G, i.e., $\ell_{G^\perp}(v) = \ell_G(v)^\perp$. For any two graphs G and H, the **implication** $G \multimap H$ is defined to be the graph $G^\perp \mathbin{⅋} H$.

Example 4.2. To give an example, consider the graph G on the left below

$$\tag{4.1}$$

Its negation $G^\perp$ is shown on the right above.

Observation 4.3. The dual graph construction defines the standard De Morgan dualities relating conjunction and disjunction, i.e., for every formula ϕ, we have $[\![\phi^\perp]\!] \simeq [\![\phi]\!]^\perp$. De Morgan dualities extended to prime graphs as $P(\!(M_1, \ldots, M_n)\!)^\perp \simeq P^\perp(\!(M_1^\perp, \ldots, M_n^\perp)\!)$, where $P^\perp$ is the dual graph of P. Furthermore, $P^\perp$ is prime if and only if P is prime. Thus each pair of prime graphs P and $P^\perp$ defines a pair of connectives that are De Morgan duals to each other.

We will now develop our proof system based on the above notion of negation as graph duality. The requirements mentioned in the introduction entail that:

(i) for any isomorphic graphs G and H, the graph $G \multimap H$ should be provable;
(ii) if $G \neq \varnothing$ then G and $G^\perp$ should not be both provable;
(iii) the implication $\multimap$ should be transitive, i.e., if $G \multimap H$, and $H \multimap K$ are provable then so should be $G \multimap K$;
(iv) the implication $\multimap$ should be closed under context, i.e., if $G \multimap H$ is provable and $C[\cdot]_R$ is an arbitrary context, then $C[G]_R \multimap C[H]_R$ should be provable;
(v) if A and C are provable graphs, and $R \subseteq V_C$, then the graph $C[A]_R$ should also be provable.

As our chosen notation suggests, our proof system will be an extension of multiplicative linear logic (MLL). Note that this is not our starting point, but a consequence of the conditions above. Since we have a common unit for $\parr$ and $\otimes$, namely the empty graph, a form of *mix* (i.e., $A \otimes B \multimap A \parr B$) will be derivable in our system. Indeed, it turns out that our proof system is a conservative extension of MLL with mix [Gir87a, Bel97, FR94]. This will be discussed in detail in Section 8.

Example 4.4. As an example, consider the following three graphs:

$$A_1: \begin{array}{cc} a^\perp & a \\ \times \\ b & b^\perp \end{array} \qquad A_2: \begin{array}{cc} a^\perp & a \\ \\ b & b^\perp \end{array} \qquad A_3: \begin{array}{cc} a^\perp & a \\ \\ b & b^\perp \end{array} \tag{4.2}$$

The graph A_1 on the left should clearly be provable, as it corresponds to the formula $(a^\perp \parr a) \otimes (b \parr b^\perp)$, which is provable in MLL. The graph A_3 on the right should not be provable, as it corresponds to the formula $(a^\perp \otimes b) \parr (a \otimes b^\perp)$, which is not provable in MLL. But what about the graph A_2 in the middle? It does not correspond to a formula, and therefore we cannot resort to MLL. Nonetheless, we can make the following observations. If A_2 were provable, then so would be the graph A_4 shown below:

$$A_4: \begin{array}{cc} a^\perp & a \\ \\ a & a^\perp \end{array} \tag{4.3}$$

as it is obtained from A_2 by a simple substitution, i.e., replacing b with a. However, $A_4^\perp \simeq A_4$, and therefore $A_4^\perp$ and A_4 would both be provable, which would be a contradiction and should be ruled out. Hence, A_2 should not be provable. It also follows that $A_1 \multimap A_2$ cannot hold, as otherwise we would be able to use A_1 and modus ponens to establish that A_2 is provable, which cannot hold as we just observed. By applying a similar argument, we conclude that $A_2 \multimap A_3$ does not hold either.

This example also shows that that implication is not simply subset inclusion of edges. However, in our minimal logic on graphs that we present here, the converse does hold: we will see later that whenever we have that $G \multimap H$ is provable and $V_G = V_H$ then $E_H \subseteq E_G$.[6]

For presenting the inference system we use a *deep inference* formalism [GS01, Gug07], which allows rewriting inside an arbitrary context and admits a rather flexible composition of derivations. In our presentation we will follow the notation of *open deduction*, introduced in [GGP10].

---

[6]But this observation is not true in general for all logics that might be designed on graphs. For example in the extension of Boolean logic, defined in [CDW20], is is not necessarily true that implication preserves edges.

Let us start with the following two inference rules

$$\mathsf{i}{\downarrow}\,\frac{\varnothing}{A^{\perp}\,\mathbin{\rotatebox[origin=c]{180}{\&}}\,A} \qquad\qquad \mathsf{s.sw}{\uparrow}\,\frac{B\otimes A}{B[A]_S}\;S\subseteq V_B,\;S\neq V_B \qquad\qquad (4.4)$$

which are induced by the two Points (i) and (v) above, and which are called **identity down** and **super switch up**, respectively. The $\mathsf{i}{\downarrow}$ says that for arbitrary graphs C and A and any $R\subseteq V_C$, if C is provable, then so is the graph $C[A\,\mathbin{\rotatebox[origin=c]{180}{\&}}\,A^{\perp}]_R$. Similarly, the rule $\mathsf{s.sw}{\uparrow}$ says that whenever $C[B\otimes A]_R$ is provable, then so is $C[B[A]_S]_R$ for any three graphs A, B, C and any $R\subseteq V_C$ and $S\subseteq V_B$. The condition $S\neq V_B$ is there to avoid a trivial rule instance, as $B[A]_S = B\otimes A$ if $S = V_B$. Practically, the rule $\mathsf{s.sw}{\uparrow}$ only removes some of the edges between the modules A and B in the graph $C[B\otimes A]_R$. But it is important to observe that we cannot simply remove arbitrary edges. We always have to ensure that A remains a module in the resulting graph. More precisely, in $B\otimes A$, both B and A are modules, but in $B[A]_S$ only A is a module, but B is not.

As the name "deep inference" suggests, we want to apply the inference rules inside any context $C[\cdot]_R$, and not only at the top level in the modular decomposition tree. For this, we define now how we build derivations from inference rules.

Definition 4.5. An **inference system** S is a set of inference rules. We define the set of **derivations** in S inductively below, and we denote a derivation $\mathcal{D}$ in S with premise G and conclusion H, as follows:

$$\begin{array}{c} G \\ \mathcal{D}\,\|\,\mathsf{S} \\ H \end{array}$$

(1) Every graph G is a derivation (also denoted by G) with premise G and conclusion G.
(2) If G is a graph with n vertices and $\mathcal{D}_1,\dots,\mathcal{D}_n$ are derivations with premise G_i and conclusion H_i for each $i\in\{1,\dots,n\}$, then $G(\!|\mathcal{D}_1,\dots,\mathcal{D}_n|\!)$ is a derivation with premise $G(\!|G_1,\dots,G_n|\!)$ and conclusion $G(\!|H_1,\dots,H_n|\!)$ denoted as

$$G\left(\!\left|\begin{array}{c} G_1 \\ \mathcal{D}_1\,\|\,\mathsf{S} \\ H_1 \end{array},\dots,\begin{array}{c} G_n \\ \mathcal{D}_n\,\|\,\mathsf{S} \\ H_n \end{array}\right|\!\right)$$

respectively. If $G = \mathbin{\rotatebox[origin=c]{180}{\&}}$ or $G = \otimes$ we may respectively write

$$\begin{array}{cc} G_1 & G_2 \\ \mathcal{D}_1\,\|\,\mathsf{S} \;\mathbin{\rotatebox[origin=c]{180}{\&}}\; \mathcal{D}_2\,\|\,\mathsf{S} \\ H_1 & H_2 \end{array} \qquad\text{or}\qquad \begin{array}{cc} G_1 & G_2 \\ \mathcal{D}_1\,\|\,\mathsf{S} \;\otimes\; \mathcal{D}_2\,\|\,\mathsf{S} \\ H_1 & H_2 \end{array}$$

(3) If $\mathcal{D}_1$ is a derivation with premise G_1 and conclusion H_1, and $\mathcal{D}_2$ is a derivation with premise G_2 and conclusion H_2, and

$$\mathsf{r}\,\frac{H_1}{G_2}$$

is an instance of an inference rule r, then $\mathcal{D}_2 \circ_r \mathcal{D}_1$ is a derivation with premise G_1 and conclusion H_2, denoted as

$$
r\frac{\begin{array}{c}G_1 \\ \mathcal{D}_1 \,\|\, \mathsf{S} \\ H_1\end{array}}{\begin{array}{c}G_2 \\ \mathcal{D}_2 \,\|\, \mathsf{S} \\ H_2\end{array}}
\qquad \text{or} \qquad
\begin{array}{c}G_1 \\ \mathcal{D}_1 \,\|\, \mathsf{S} \\ r\dfrac{H_1}{G_2} \\ \mathcal{D}_2 \,\|\, \mathsf{S} \\ H_2\end{array}
\qquad \text{or} \qquad
\begin{array}{c}G_1 \\ \mathcal{D}_1 \,\|\, \mathsf{S} \\ r\dfrac{H_1}{G_2} \\ \mathcal{D}_2 \,\|\, \mathsf{S} \\ H_2\end{array}
$$

If $H_1 \simeq_f G_2$ we can compose $\mathcal{D}_1$ and $\mathcal{D}_2$ directly to $\mathcal{D}_2 \circ \mathcal{D}_1$, denoted as

$$
\begin{array}{c}G_1 \\ \mathcal{D}_1 \,\|\, \mathsf{S} \\ H_1 \\ f\, {\cdots\cdots} \\ G_2 \\ \mathcal{D}_2 \,\|\, \mathsf{S} \\ H_2\end{array}
\ \text{or} \
\begin{array}{c}G_1 \\ \mathcal{D}_1 \,\|\, \mathsf{S} \\ H_1 \\ \simeq\, {\cdots\cdots} \\ G_2 \\ \mathcal{D}_2 \,\|\, \mathsf{S} \\ H_2\end{array}
\ \text{or} \
\begin{array}{c}G_1 \\ \mathcal{D}_1 \,\|\, \mathsf{S} \\ H_1 \\ {\cdots\cdots} \\ G_2 \\ \mathcal{D}_2 \,\|\, \mathsf{S} \\ H_2\end{array}
\ \text{or} \
\begin{array}{c}G_1 \\ \mathcal{D}_1 \,\|\, \mathsf{S} \\ H_1 \\ r\, {\cdots} \\ G_2 \\ \mathcal{D}_2 \,\|\, \mathsf{S} \\ H_2\end{array}
\ \text{or} \
\begin{array}{c}G_1 \\ \mathcal{D}_1 \,\|\, \mathsf{S} \\ H_1 \\ r\, {\cdots} \\ G_2 \\ \mathcal{D}_2 \,\|\, \mathsf{S} \\ H_2\end{array}
\qquad (4.5)
$$

If f is clear from context we will simply write $\mathcal{D}_2 \circ \mathcal{D}_1$ as

$$
\begin{array}{c}G_1 \\ \mathcal{D}_1 \,\|\, \mathsf{S} \\ H_1 \\ \mathcal{D}_2 \,\|\, \mathsf{S} \\ H_2\end{array}
\qquad \text{or} \qquad
\begin{array}{c}G_1 \\ \mathcal{D}_1 \,\|\, \mathsf{S} \\ G_2 \\ \mathcal{D}_2 \,\|\, \mathsf{S} \\ H_2\end{array}
\qquad (4.6)
$$

where we only write H_1 or G_2 in the middle, omitting the isomorphism step, in order to ease readability. However, even if f is not written, we always assume it is part of the derivation and explicitly given.

A **proof** in S is a derivation in S whose premise is $\varnothing$. A graph G is **provable** in S iff there is a proof in S with conclusion G. We denote this as $\vdash_\mathsf{S} G$ (or simply as $\vdash G$ if S is clear from the context). The **length** of a derivation $\mathcal{D}$, denoted by $|\mathcal{D}|$, is the number of inference rule instances in $\mathcal{D}$.

Remark 4.6. If we have a derivation $\mathcal{D}$ from A to B, and a context $G[\cdot]_R$, then we also have a derivation from $G[A]_R$ to $G[B]_R$. We can write this derivation as

$$
\begin{array}{c}G[A]_R \\ G[\mathcal{D}]_R\,\| \\ G[B]_R\end{array}
\qquad \text{or} \qquad
G\left[\begin{array}{c}A \\ \mathcal{D}\,\| \\ B\end{array}\right]_R
$$

If $\mathcal{D}$ consists of a single inference rule, we sometimes write (with a slight abuse of notation)

$$
G\left[\,r\dfrac{A}{B}\,\right]_R
\qquad \text{as} \qquad
r\dfrac{G[A]_R}{G[B]_R}
\qquad (4.7)
$$

Remark 4.7. In order to ease readability of derivations, we will mostly omit the isomorphism steps shown in (4.5), and use the abbreviation (4.6). This concerns also situations as shown below left, which are written as shown below right:

$$
\simeq
\begin{array}{c}
\mathsf{r_1}\dfrac{A}{B} \\[2pt]
\vdots \\[-2pt]
\mathsf{r_2}\dfrac{B'}{C}
\end{array}
\qquad \rightsquigarrow \qquad
\mathsf{r_2}\dfrac{\mathsf{r_1}\dfrac{A}{B}}{C}
\quad \text{or} \quad
\mathsf{r_2}\dfrac{\mathsf{r_1}\dfrac{A}{B'}}{C}
\tag{4.8}
$$

In other words, in derivations, we consider graphs equal modulo isomorphisms. In particular, for the length $|\mathcal{D}|$ of a derivation, we count only the inference steps that are not isomorphisms. This abuse of notation is justified as these isomorphisms can be permuted with all inference rules that we discuss in this paper. Furthermore, when we draw the graphs as in (4.9) below, the isomorphisms are not visible.

Example 4.8. The following derivation is an example of a proof of length 2, using only i↓ and s.sw↑:

$$\tag{4.9}$$

where the s.sw↑ instance moves the module d in the context consisting of vertices labelled a, b, c. Let us emphasise that the conclusion of the derivation in (4.9) is not a formula but a graph. It establishes that the following implication is provable:

$$\tag{4.10}$$

which is a fact beyond the scope of formulas.

Remark 4.9. In [GGP10] it is shown how a derivation on formulas in the style of open deduction (as we defined it in Definition 4.5), can be translated into a derivation in the style of the calculus of structures [Gug07, GS01], which can be seen as a sequence of rewriting steps on formulas. The same also works on derivations for graphs. To give an example, consider the derivation on the left below:

$$\tag{4.11}$$

which can also be written as shown on the right above. (Note that we make use of the notation in (4.7)). In [GGP10], the derivation on the left in (4.11) is called ***synchronal***

and the one on the right in (4.11) is called **sequential**. For every synchronal derivation there is at least one sequential variant (but it is not unique). In later parts of the paper we sometimes speak of the "bottommost" rule instance in a derivation. In these cases, we assume that we have fixed some sequential form. Observe that both derivations in (4.11) above make use of (4.8).

As in other deep inference systems, we can give for the rules in (4.4) their *duals*. In general, if

$$r\,\frac{G}{H}$$

is an instance of a rule, then

$$r^\perp\,\frac{H^\perp}{G^\perp}$$

is an instance of the dual rule. The duals of the two rules in (4.4) are the following:

$$\mathsf{i{\uparrow}}\,\frac{A \otimes A^\perp}{\varnothing} \qquad\qquad \mathsf{s.sw{\downarrow}}\,\frac{B[A]_S}{B \,\mathbin{\rotatebox[origin=c]{180}{\&}}\, A}\ S \subseteq V_B,\ S \neq \varnothing \tag{4.12}$$

called **identity up** (or **cut**) and **super switch down**, respectively. We have the side condition $S \neq \varnothing$ to avoid a triviality, as $B[A]_S = B \,\mathbin{\rotatebox[origin=c]{180}{\&}}\, A$ if $S = \varnothing$. Dually to s.sw↑, the s.sw↓-rule removes all the edges between A and B.

Example 4.10. The implication in (4.10) can also be proven using only only s.sw↓ and i↓ instead of s.sw↑ and i↓, as the following proof of length 3 shows:

$$\tag{4.13}$$

Definition 4.11. Let S be an inference system. We say that an inference rule r is **derivable** in S iff

$$\text{for every instance }\ r\,\frac{G}{H}\quad\text{there is a derivation}\quad \mathcal{D}\,\Big\|\mathsf{s}\ \begin{matrix}G\\[2pt] \\ H\end{matrix}\ .$$

We say that r is **admissible** in S iff

$$\text{for every instance }\ r\,\frac{G}{H}\quad\text{we have that }\vdash_\mathsf{S} G\text{ implies }\vdash_\mathsf{S} H\ .$$

If r ∈ S then r is trivially derivable and admissible in S.

Most deep inference systems in the literature (e.g. [GS01, BT01, Gug07, Str02, GS02, HTAC19]) contain the **switch** rule:[7]

$$\mathsf{sw}\frac{(A \bindnasrepma B) \otimes C}{A \bindnasrepma (B \otimes C)} \tag{4.14}$$

One can immediately see that it is its own dual and is a special case of both $\mathsf{s.sw}{\downarrow}$ and $\mathsf{s.sw}{\uparrow}$. We therefore have the following:

Lemma 4.12. *If in an inference system S one of the rules $\mathsf{s.sw}{\downarrow}$ and $\mathsf{s.sw}{\uparrow}$ is derivable, then so is sw.*

Remark 4.13. In a standard deep inference system for formulas we also have the converse of Lemma 4.12, i.e., if sw is derivable, then so are $\mathsf{s.sw}{\uparrow}$ and $\mathsf{s.sw}{\downarrow}$ (see, e.g., [Str03a]). However, in the case of arbitrary graphs this is no longer true, and the rules $\mathsf{s.sw}{\uparrow}$ and $\mathsf{s.sw}{\downarrow}$ are strictly more powerful than sw.

Lemma 4.14. *Let S be an inference system. If the rules $\mathsf{i}{\downarrow}$ and $\mathsf{i}{\uparrow}$ and sw are derivable in S, then for every rule r that is derivable in S, also its dual $\mathsf{r}^{\perp}$ is derivable in S.*

Proof. Suppose we have two graphs G and H, and a derivation from G to H in S. Then it suffices to show that we can construct a derivation from $H^{\perp}$ to $G^{\perp}$ in S:

$$\mathsf{sw}\frac{\mathsf{i}{\downarrow}\dfrac{\varnothing}{G^{\perp} \bindnasrepma G} \otimes H^{\perp}}{G^{\perp} \bindnasrepma \dfrac{\begin{array}{c} G \\ \|\mathsf{s} \\ H \end{array} \otimes H^{\perp}}{\mathsf{i}{\uparrow}\dfrac{}{\varnothing}}}$$

Note that $\varnothing \otimes H^{\perp} = H^{\perp}$ and $G^{\perp} \bindnasrepma \varnothing = G^{\perp}$.   □

Lemma 4.15. *If the rules $\mathsf{i}{\uparrow}$ and sw are admissible for an inference system S, then $\multimap$ is transitive, i.e., if $\vdash_{\mathsf{S}} G \multimap H$ and $\vdash_{\mathsf{S}} H \multimap K$ then $\vdash_{\mathsf{S}} G \multimap K$.*

Proof. We can construct the following derivation

$$\mathsf{sw}\frac{\begin{array}{c} \varnothing \\ \|\mathsf{s} \\ G^{\perp} \bindnasrepma H \end{array} \otimes \begin{array}{c} \varnothing \\ \|\mathsf{s} \\ H^{\perp} \bindnasrepma K \end{array}}{G^{\perp} \bindnasrepma \mathsf{sw}\dfrac{H \otimes (H^{\perp} \bindnasrepma K)}{\mathsf{i}{\uparrow}\dfrac{H \otimes H^{\perp}}{\varnothing} \bindnasrepma K}} \tag{4.15}$$

from $\varnothing$ to $G^{\perp} \bindnasrepma K$ in S.   □

---

[7]This rule has also been used in [Ret99] and [Ret03], and is also known under the names *weak distributivity* [BCST96] or *dissociativity* [DP04].

Lemma 4.15 is the reason why i↑ is also called *cut*. In a well-designed deep inference system for formulas, the two rules i↓ and i↑ can be restricted in a way that they are only applicable to atoms, i.e., replaced by the following two rules that we call **atomic identity down** and **atomic identity up**, respectively:

$$\mathsf{ai}{\downarrow}\frac{\varnothing}{a^{\perp} \,\mathfrak{N}\, a} \qquad \text{and} \qquad \mathsf{ai}{\uparrow}\frac{a \otimes a^{\perp}}{\varnothing} \tag{4.16}$$

We would like to achieve something similar for our proof system on graphs. For this it is necessary to be able to decompose prime graphs into atoms, but the two rules s.sw↓ and s.sw↑ cannot do this, as they are only able to move around modules in a graph. For this reason, we add the following two rules to our system:

$$\mathsf{p}{\downarrow}\frac{(M_1 \,\mathfrak{N}\, N_1) \otimes \cdots \otimes (M_n \,\mathfrak{N}\, N_n)}{P(\!|M_1, \ldots, M_n|\!) \,\mathfrak{N}\, P^{\perp}(\!|N_1, \ldots, N_n|\!)} \quad P \text{ prime, } |V_P|{\geq}4, \ M_1{\neq}\varnothing,...,M_n{\neq}\varnothing \tag{4.17}$$

called **prime down**, and

$$\mathsf{p}{\uparrow}\frac{P(\!|M_1, \ldots, M_n|\!) \otimes P^{\perp}(\!|N_1, \ldots, N_n|\!)}{(M_1 \otimes N_1) \,\mathfrak{N}\, \cdots \,\mathfrak{N}\, (M_n \otimes N_n)} \quad P \text{ prime, } |V_P|{\geq}4 \ M_1{\neq}\varnothing,...,M_n{\neq}\varnothing \tag{4.18}$$

called **prime up**. In both cases, the side condition is that P needs to be a prime graph and has at least 4 vertices. We also require that for all $i \in \{1, \ldots, n\}$ we have that M_i is non-empty in an application of p↓ and p↑.[8] The reason for the side conditions on the p↓ is not that the rules would become unsound otherwise. In fact, these rules without side conditions are admissible in the general case. The details will be discussed in the proof of Lemma 5.1 and at the end of Section 7.

Example 4.16. Below is a derivation of length 5 using the p↓-rule, and proving that a prime graph implies itself. On the left, we use the "formula style" notation of the modular decomposition of graphs, as in (3.5), and on the right we use the graphical notation, as in (3.3) and (3.4).

---

[8]In the conference version [AHS20] of this paper, we had the weaker condition that for all $i \in \{1, \ldots, n\}$ at least one of M_i and N_i is non-empty. The advantage of the stronger condition that we use here is that we obtain a more controlled proof system as the prime graphs involved in a p↓ can only be those that appear in the modular decomposition of the graph.

SGS

GS

$$\mathsf{ai}{\downarrow}\ \frac{\varnothing}{a^{\perp}\ \mathbin{⅋}\ a} \qquad\qquad \mathsf{s.sw}{\downarrow}\ \frac{B[A]_S}{B\ \mathbin{⅋}\ A}\ \ S\subseteq V_B,\ S\neq\varnothing,\ A\neq\varnothing$$

$$\mathsf{p}{\downarrow}\ \frac{(M_1\ \mathbin{⅋}\ N_1)\otimes\cdots\otimes(M_n\ \mathbin{⅋}\ N_n)}{P^{\perp}(\!|M_1,\ldots,M_n|\!)\ \mathbin{⅋}\ P(\!|N_1,\ldots,N_n|\!)}\ \ P\ \text{prime},\ |V_P|{\geq}4,\ M_1{\neq}\varnothing,\ ...,\ M_n{\neq}\varnothing$$

$$\mathsf{ai}{\uparrow}\ \frac{a\otimes a^{\perp}}{\varnothing} \qquad\qquad \mathsf{s.sw}{\uparrow}\ \frac{B\otimes A}{B[A]_S}\ \ S\subseteq V_B,\ S\neq V_B,\ A\neq\varnothing$$

$$\mathsf{p}{\uparrow}\ \frac{P(\!|M_1,\ldots,M_n|\!)\otimes P^{\perp}(\!|N_1,\ldots,N_n|\!)}{(M_1\otimes N_1)\ \mathbin{⅋}\cdots\mathbin{⅋}\ (M_n\otimes N_n)}\ \ P\ \text{prime},\ |V_P|{\geq}4,\ M_1{\neq}\varnothing,\ ...,\ M_n{\neq}\varnothing$$

Figure 1: The inference rules for systems SGS and GS.

But let us emphasize, that in both cases it is the same derivation. We could also use a mixed notation, as shown below.

This completes the presentation of our system, which is shown in Figure 1, and formally defined below.

Definition 4.17. We define *system* **SGS** to be the set $\{\mathsf{ai}{\downarrow}, \mathsf{s.sw}{\downarrow}, \mathsf{p}{\downarrow}, \mathsf{p}{\uparrow}, \mathsf{s.sw}{\uparrow}, \mathsf{ai}{\uparrow}\}$ of inference rules shown in Figure 1. The *down-fragment* (resp. *up-fragment*) of SGS consists of the rules $\{\mathsf{ai}{\downarrow}, \mathsf{s.sw}{\downarrow}, \mathsf{p}{\downarrow}\}$ (resp. $\{\mathsf{ai}{\uparrow}, \mathsf{s.sw}{\uparrow}, \mathsf{p}{\uparrow}\}$) and is denoted by SGS$\downarrow$ (resp. SGS$\uparrow$). The down-fragment SGS$\downarrow$ is also called *system* **GS**.

5. Properties of the Proof System

The properties that we establish here are standard for deep inference systems (see, e.g., [GS01, TS19]). The first observation about SGS is that the general forms of the identity rules i$\downarrow$ and i$\uparrow$ are derivable, as we will see in Corollary 5.2 below. Before, we show an auxiliary lemma stating the corresponding result for the prime rules, namely that their general form is also derivable, i.e., they can be applied to any graph instead of only prime graphs.

Lemma 5.1. *Let G and $M_1, \ldots, M_n, N_1, \ldots, N_n$ be graphs with $|V_G| = n$, such that for every $i \in \{1, \ldots, n\}$, we have that $M_i = \varnothing$ implies $N_i = \varnothing$. Then there are derivations*

$$
\begin{array}{c}
(M_1 \,\bindnasrepma\, N_1) \otimes \cdots \otimes (M_n \,\bindnasrepma\, N_n) \\
\| \, \mathsf{SGS}{\downarrow} \\
G(\!|M_1, \ldots, M_n|\!) \,\bindnasrepma\, G^\perp(\!|N_1, \ldots, N_n|\!)
\end{array}
\tag{5.1}
$$

and dually

$$
\begin{array}{c}
G(\!|M_1, \ldots, M_n|\!) \otimes G^\perp(\!|N_1, \ldots, N_n|\!) \\
\| \, \mathsf{SGS}{\uparrow} \\
(M_1 \otimes N_1) \,\bindnasrepma\, \cdots \,\bindnasrepma\, (M_n \otimes N_n)
\end{array}
\tag{5.2}
$$

Proof. For showing (5.1), we proceed by induction on $|V_G|$. First consider the case that there is an $i \in \{1, \ldots, n\}$ such that $N_i = \varnothing = M_i$. Without loss of generality, we can assume that $i = 1$. Then there is a graph H with $|V_H| = n - 1$ and $G(\!|M_1, \ldots, M_n|\!) = H(\!|M_2, \ldots, M_n|\!)$ and $G^\perp(\!|N_1, \ldots, N_n|\!) = H^\perp(\!|N_2, \ldots, N_n|\!)$, and therefore we have the derivation

$$
\simeq \frac{
\begin{array}{c}
(\varnothing \,\bindnasrepma\, \varnothing) \otimes (M_2 \,\bindnasrepma\, N_2) \otimes \cdots \otimes (M_n \,\bindnasrepma\, N_n) \\
\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\
(M_2 \,\bindnasrepma\, N_2) \otimes \cdots \otimes (M_n \,\bindnasrepma\, N_n) \\
\mathcal{D} \| \, \mathsf{SGS}{\downarrow} \\
H(\!|M_2, \ldots, M_n|\!) \,\bindnasrepma\, H^\perp(\!|N_2, \ldots, N_n|\!) \\
\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots
\end{array}
}{
G(\!|\varnothing, M_2, \ldots, M_n|\!) \,\bindnasrepma\, G^\perp(\!|\varnothing, N_2, \ldots, N_n|\!)
}
$$

where $\mathcal{D}$ exists by induction hypothesis. We can therefore now assume that $M_i \neq \varnothing$ for all $i \in \{1, \ldots, n\}$. We now make a case analysis on G using Lemma 3.9:

(i) If G is a singleton graph, the statement holds trivially.

(ii) If $G = A \otimes B$ then $G(\!|M_1, \ldots, M_n|\!) = A(\!|M_1, \ldots, M_k|\!) \otimes B(\!|M_{k+1}, \ldots, M_n|\!)$ and $G^\perp(\!|N_1, \ldots, N_n|\!) = A^\perp(\!|N_1, \ldots, N_k|\!) \,\bindnasrepma\, B^\perp(\!|N_{k+1}, \ldots, N_n|\!)$ for some $1 \leq k \leq n$. We therefore have

$$
\mathsf{s.sw}{\downarrow} \frac{
\left[
\begin{array}{c}
(M_1 \,\bindnasrepma\, N_1) \otimes \cdots \otimes (M_k \,\bindnasrepma\, N_k) \\
\mathcal{D}_1 \| \, \mathsf{SGS}{\downarrow} \\
A(\!|M_1, \ldots, M_k|\!) \,\bindnasrepma\, A^\perp(\!|N_1, \ldots, N_k|\!)
\end{array}
\otimes
\begin{array}{c}
(M_{k+1} \,\bindnasrepma\, N_{k+1}) \otimes \cdots \otimes (M_n \,\bindnasrepma\, N_n) \\
\mathcal{D}_2 \| \, \mathsf{SGS}{\downarrow} \\
B(\!|M_{k+1}, \ldots, M_n|\!) \,\bindnasrepma\, B^\perp(\!|N_{k+1}, \ldots, N_n|\!)
\end{array}
\right]
}{
\mathsf{s.sw}{\downarrow} \dfrac{(A(\!|M_1, \ldots, M_k|\!) \,\bindnasrepma\, A^\perp(\!|N_1, \ldots, N_k|\!)) \otimes B(\!|M_{k+1}, \ldots, M_n|\!)}{(A(\!|M_1, \ldots, M_k|\!) \otimes B(\!|M_{k+1}, \ldots, M_n|\!)) \,\bindnasrepma\, A^\perp(\!|N_1, \ldots, N_k|\!)} \,\bindnasrepma\, B^\perp(\!|N_{k+1}, \ldots, N_n|\!)
}
$$

where $\mathcal{D}_1$ and $\mathcal{D}_2$ exist by induction hypothesis. Note that it is possible that $A^\perp(\!|N_1, \ldots, N_k|\!) = \varnothing$ or $B^\perp(\!|N_{k+1}, \ldots, N_n|\!) = \varnothing$ (or both). In that case one (or both) of the two instances of $\mathsf{s.sw}{\downarrow}$ is vacuous.

(iii) If $G = A \,\bindnasrepma\, B$, we proceed similarly.

(iv) If $G = P(\!|A_1, \ldots, A_k|\!)$ for P prime and $k = |V_P| \geq 4$ and $A_l \neq \varnothing$ for each $l \in \{1, \ldots, k\}$, then we have that $G(\!|M_1, \ldots, M_n|\!) = P(\!|A_1(\!|M_{11}, \ldots, M_{1h_1}|\!), \ldots, A_k(\!|M_{k1}, \ldots, M_{kh_k}|\!)|\!)$ where $\{M_1, \ldots, M_n\} = \mathcal{M}_1 \cup \cdots \cup \mathcal{M}_k$ and $\mathcal{M}_l = \{M_{l1}, \ldots, M_{lh_l}\}$ and $\mathcal{M}_l \cap \mathcal{M}_j = \varnothing$ for $l \neq j$. Similarly, we have $G^\perp(\!|N_1, \ldots, N_n|\!) = P^\perp(\!|A_1^\perp(\!|N_{11}, \ldots, N_{1h_1}|\!), \ldots, A_k^\perp(\!|N_{k1}, \ldots, N_{kh_k}|\!)|\!)$ where $\{N_1, \ldots, N_n\} = \mathcal{N}_1 \cup \cdots \cup \mathcal{N}_k$ and $\mathcal{N}_l = \{N_{l1}, \ldots, N_{lh_l}\}$ and $\mathcal{N}_l \cap \mathcal{N}_j = \varnothing$ for $l \neq j$. Since $M_i \neq \varnothing$ for all $i \in \{1, \ldots, n\}$,

we also have that $A_l(\!|M_{l1}, \ldots, M_{lh_l}|\!) \neq \varnothing$ for each $l \in \{1, \ldots, k\}$. Therefore, we have the following derivation

$$
\cfrac{
\cfrac{(M_{11} \,\invamp\, N_{11}) \otimes \cdots \otimes (M_{1h_1} \,\invamp\, N_{1h_1})}{A_1^\perp(\!|M_{11}, \ldots, M_{1h_1}|\!) \,\invamp\, A_1(\!|N_{11}, \ldots, N_{1h_1}|\!)} {\,\mathcal{D}_1\,\|\,\mathsf{SGS}{\downarrow}} \;\otimes\cdots\otimes\;
\cfrac{(M_{k1} \,\invamp\, N_{k1}) \otimes \cdots \otimes (M_{kh_k} \,\invamp\, N_{kh_k})}{A_k^\perp(\!|M_{k1}, \ldots, M_{kh_k}|\!) \,\invamp\, A_k(\!|N_{k1}, \ldots, N_{kh_k}|\!)} {\,\mathcal{D}_k\,\|\,\mathsf{SGS}{\downarrow}}
}{P(\!|A_1(\!|M_{11}, \ldots, M_{1h_1}|\!), \ldots, A_k(\!|M_{k1}, \ldots, M_{kh_k}|\!)|\!) \,\invamp\, P^\perp(\!|A_1^\perp(\!|N_{11}, \ldots, N_{1h_1}|\!), \ldots, A_k^\perp(\!|N_{k1}, \ldots, N_{kh_k}|\!)|\!)} \mathsf{p}{\downarrow}
$$

where $\mathcal{D}_1, \ldots, \mathcal{D}_k$ exist by induction hypothesis.

The derivation in (5.2) can be constructed dually. $\square$

Corollary 5.2. *The rule* i↓ *is derivable in* SGS↓*, and dually,* i↑ *is derivable in* SGS↑*.*

Proof. Each graph G, can be written as $G = G(\!|a_1, \ldots, a_n|\!)$ where $V_G = \{a_1, \ldots, a_n\}$. Since $a_1, \ldots, a_n$ are non-empty modules of G we can apply Lemma 5.1 to obtain the following derivation of $G \,\invamp\, G^\perp$ from $\varnothing$ in SGS↓:

$$
\cfrac{
\cfrac{\varnothing}{a_1 \,\invamp\, a_1^\perp}\mathsf{ai}{\downarrow} \;\otimes\cdots\otimes\; \cfrac{\varnothing}{a_n \,\invamp\, a_n^\perp}\mathsf{ai}{\downarrow}
}{G(\!|a_1, \ldots, a_n|\!) \,\invamp\, G^\perp(\!|a_1^\perp, \ldots, a_n^\perp|\!)} {\,\mathcal{D}\,\|\,\mathsf{SGS}{\downarrow}}
\tag{5.3}
$$

where $\mathcal{D}$ exists by Lemma 5.1. Dually, we can show that for any G there is a derivation from $G \otimes G^\perp$ to $\varnothing$ in SGS↑. $\square$

Corollary 5.3. *If* $G, M_1, \ldots, M_n$ *are non-empty graphs, then there is a derivation*

$$
\cfrac{M_1 \otimes \cdots \otimes M_n}{G(\!|M_1, \ldots, M_n|\!)} {\,\|\,\mathsf{GS}}
$$

Proof. This follows from Lemma 5.1, using $G^\perp(\!|\varnothing, \ldots, \varnothing|\!)$. $\square$

Next, observe that Lemmas 4.14 and 4.15 hold for system SGS. In particular, we have that if $\vdash_{\mathsf{SGS}} A \multimap B$ and $\vdash_{\mathsf{SGS}} B \multimap C$ then $\vdash_{\mathsf{SGS}} A \multimap C$ since i↑ $\in$ SGS. The main result of this paper is that Lemma 4.15 also holds for GS. More precisely, we have the following theorem:

Theorem 5.4 (Cut Admissibility). *The rule* i↑ *is admissible for* GS*.*

To prove this theorem, we will show that the whole up-fragment of SGS is admissible for GS.

Theorem 5.5. *The rules* ai↑, s.sw↑, p↑ *are admissible for* GS*.*

Then Theorem 5.4 follows immediately from Theorem 5.5 and the second statement in Corollary 5.2.

The following two sections are devoted to the proof of Theorem 5.5. But before, let us finish this section by exhibiting some immediate consequences of Theorem 5.4.

Corollary 5.6. *For every graph G, we have* $\vdash_{\mathsf{SGS}} G$ *iff* $\vdash_{\mathsf{GS}} G$*.*

Corollary 5.7. *For all graphs G and H, we have*

$$\vdash_{\mathsf{GS}} G \multimap H \iff \begin{array}{c} \varnothing \\ \| \, \mathsf{GS} \\ G^\perp \,\bindnasrepma\, H \end{array} \iff \begin{array}{c} \varnothing \\ \| \, \mathsf{SGS} \\ G^\perp \,\bindnasrepma\, H \end{array} \iff \begin{array}{c} G \\ \| \, \mathsf{SGS} \\ H \end{array}$$

Proof. The first equivalence is just the definition of $\vdash$. The second equivalence follows from Theorem 5.5, and the last equivalence follows from the two derivations

$$\mathsf{i}{\downarrow}\,\frac{\varnothing}{G^\perp \,\bindnasrepma\, \begin{array}{c} G \\ \| \\ H \end{array}} \quad \text{and} \quad \mathsf{s.sw}{\downarrow}\,\frac{G \otimes \begin{array}{c} \varnothing \\ \| \\ G^\perp \,\bindnasrepma\, H \end{array}}{\mathsf{i}{\uparrow}\,\dfrac{G \otimes G^\perp}{\varnothing} \,\bindnasrepma\, H}$$

together with Corollary 5.2. □

Corollary 5.8. *For all graphs G and H and all contexts $C[\cdot]_R$, we have that*

$$\vdash_{\mathsf{GS}} G \multimap H \qquad \Longrightarrow \qquad \vdash_{\mathsf{GS}} C[G]_R \multimap C[H]_R \quad .$$

Proof. This is a consequence of Corollary 5.7 and Remark 4.6. But it can also be proved directly using Lemma 5.1. □

Corollary 5.9. *We have $\vdash_{\mathsf{GS}} A \otimes B \iff \vdash_{\mathsf{GS}} A$ and $\vdash_{\mathsf{GS}} B$.*

Proof. This follows immediately by inspecting the inference rules of GS. □

Corollary 5.10. *We have $\vdash_{\mathsf{GS}} P(\!|M_1, \ldots, M_n|\!)$ with P prime and $n \geq 4$ and $M_i \neq \varnothing$ for all $i = \{1, \ldots, n\}$, if and only if there is at least one $i = \{1, \ldots, n\}$ such that $\vdash_{\mathsf{GS}} M_i$ and $\vdash_{\mathsf{GS}} P(\!|M_1, \ldots, M_{i-1}, \varnothing, M_{i+1}, \ldots, M_n|\!)$.*

Proof. As before, this follows immediately by inspecting the inference rules of GS. In fact, this can be seen as a generalisation of the previous corollary. □

Corollary 5.11 (consistency). *If $G \neq \varnothing$ and $\vdash_{\mathsf{GS}} G$, then $\nvdash_{\mathsf{GS}} G^\perp$.*

Proof. Since $G \neq \varnothing$ and $\vdash_{\mathsf{GS}} G$, we have for some a that $\begin{array}{c} \mathsf{ai}{\downarrow}\,\dfrac{\varnothing}{a^\perp \,\bindnasrepma\, a} \\ \| \, \mathsf{GS} \\ G \end{array}$. Hence, by Corollary 5.7 we have $\vdash_{\mathsf{GS}} G^\perp \multimap a \otimes a^\perp$. Thus, if we assume by way of contradiction that $\vdash_{\mathsf{GS}} G^\perp$ holds then by Theorem 5.4 we have $\vdash_{\mathsf{GS}} a \otimes a^\perp$, which is impossible. □

Observation 5.12. The system GS forms a proof system in the sense of Cook and Reckhow [CR79], as the time complexity of checking the correct application of each inference step is polynomial. Note that the modular decomposition of graphs can be obtained in linear time [MS94]. Furthermore, the use of the graph isomorphisms for the composition of derivations, as in (4.5), does not increase complexity, as we assume that the isomorphism f is explicitly given, such that the correctness of the operation can also be checked in linear time.

Let us now define the notion of *size* of a graph that will play a central role in the normalisation proof as an induction measure.

Definition 5.13. The *size* of a graph G, denoted by $\|G\|$, is the lexicographic pair $\langle |V_G|, |E_{G^\perp}| \rangle$. That is, if G and H are graphs, then $\|G\| < \|H\|$ if $|V_G| < |V_H|$ or if $|V_G| = |V_H|$ and $|E_{G^\perp}| < |E_{H^\perp}|$.

Observation 5.14. Every inference rule discussed so far, except for ai↑ and i↑ do strictly decrease the size of a graph when going bottom-up in a derivation. That is, whenever we have an instance $\mathsf{r} \dfrac{H}{G}$ of an inference rule $\mathsf{r} \in \{\mathsf{ai{\downarrow}}, \mathsf{i{\downarrow}}, \mathsf{s.sw{\downarrow}}, \mathsf{s.sw{\uparrow}}, \mathsf{p{\downarrow}}, \mathsf{p{\uparrow}}, \mathsf{g{\downarrow}}, \mathsf{g{\uparrow}}\}$, then $\|H\| < \|G\|$. Since each such inference step either reduces the number of vertices or the number of edges in the dual graph (or both), and these numbers are never increased, the length of a derivation in GS with conclusion G is bound by $n^2 + n$ where $n = |V_G|$ is the number of vertices in G.[9]

Furthermore, to each graph only finitely many different non-trivial inference steps can be applied, and every derivation can be reorganised such that between any two (non-trivial) inference steps (see Remark 4.9) there is at most one isomorphism step. Hence, it follows immediately that provability in GS is decidable and in **NP**. In Section 8 we are going to show that it is also **NP**-hard, and therefore **NP**-complete.

6. Splitting and Context Reduction

We are now ready to present the fundamental properties of the system GS, namely the *splitting lemmas* and *context reduction*, that allow to prove cut elimination in the next section.[10] First, recall that the standard syntactic method for proving cut elimination in the sequent calculus is to permute the cut rule upwards in the proof, while decomposing the cut formula along its main connective, and so inductively reduce the cut rank. However, in systems formulated using deep inference this method cannot be applied, as derivations can be constructed in a more flexible way than in the sequent calculus. For this reason, the *splitting* technique has been developed in the literature on deep inference [Gug07, Str03a, GS11, HTAC19, AT17]. In this section we discuss how to prove the splitting and the context reduction lemmas for our system and how their proofs differs form the ones in the literature. The detailed proofs of the statement in this section can be found in Appendix A.

From sequent calculus to splitting for graphs. Consider the typical rule for $\otimes$ in the sequent calculus.

$$\otimes \frac{\Gamma, \phi \qquad \Delta, \psi}{\Gamma, \phi \otimes \psi, \Delta}$$

The effect of the above rule can be simulated by applying the following splitting lemma for the prime graph $\otimes$.

---

[9]For the case that the derivation contains only cographs, it can in fact be shown that the length is bound by $\frac{1}{2}n^2$ (see [BS17]), but as the details are not needed for this paper, we leave them to the reader.

[10]Proceeding via splitting and context reduction is not the only method to prove cut elimination for a deep inference proof system (see e.g. [Brü03, Str03b, AG21]), but it is the most organized and most universal method, and at the current state of art, it seems to be the only one that is applicable to GS.

Lemma 6.1 (Splitting Tensor). *Let G, A and B be graphs. If $\vdash_{\mathsf{GS}} G \,\mathbin{\mathscr{V}}\, (A \otimes B)$ then there is a context C and there are graphs K_A and K_B such that there are derivations*

$$
\begin{array}{c}
C[K_A \,\mathbin{\mathscr{V}}\, K_B]_R \\
\mathcal{D}_G \,\|\, \mathsf{GS} \\
G
\end{array}
\quad , \quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_A \,\|\, \mathsf{GS} \\
K_A \,\mathbin{\mathscr{V}}\, A
\end{array}
\quad , \quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_B \,\|\, \mathsf{GS} \\
K_B \,\mathbin{\mathscr{V}}\, B
\end{array}
\quad and \quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_C \,\|\, \mathsf{GS} \\
C
\end{array}
\quad .
$$

The graphs A and B in Lemma 6.1 play the role of formulas ϕ and ψ in the sequent calculus rule $\otimes$; while K_A and K_B play the role of sequents Γ and Δ. Thus, when we say in Lemma 6.1 that $\vdash_{\mathsf{GS}} K_A \,\mathbin{\mathscr{V}}\, A$ and $\vdash_{\mathsf{GS}} K_B \,\mathbin{\mathscr{V}}\, B$ hold, this corresponds to the provability of the premises Γ, ϕ and Δ, ψ in the sequent calculus $\otimes$-rule. This means that Lemma 6.1 allows us to rewrite a *sequent-like graph* $G \,\mathbin{\mathscr{V}}\, (A \otimes B)$ into a form where the effect of the $\otimes$-rule of the sequent calculus can be simulated.

Remark 6.2. Note that this splitting is fundamentally different from the well-known *splitting tensor lemma* for MLL proof nets [Gir87a, DR89, Ret96]. Whereas the MLL splitting tensor lemma says that *there exists* at least one tensor that is splitting, our splitting tensor lemma says that *every* tensor is splitting.[11]

Let us use the derivation in (4.13), recalled below in (6.1) for convenience, as an example to illustrate this pattern. Observe that the conclusion is of the form $G \,\mathbin{\mathscr{V}}\, ((a^\perp \,\mathbin{\mathscr{V}}\, c^\perp) \otimes b^\perp)$ for some graph G, and that after applying s.sw$\downarrow$ and ai$\downarrow$ we have split the graph G into two graph $K_A = a \otimes c$ and $K_B = b$, such that $K_A \,\mathbin{\mathscr{V}}\, a^\perp \,\mathbin{\mathscr{V}}\, c^\perp$ and $K_B \,\mathbin{\mathscr{V}}\, b^\perp$ are provable.

(6.1)

Thus applying Lemma 6.1 to the sub-graph $(a^\perp \,\mathbin{\mathscr{V}}\, c^\perp) \otimes b^\perp$ above leads us to proof (6.1).

There is a crucial difference between the statement of Lemma 6.1 and other splitting lemmas in the in the literature on deep inference [Gug07, Str03a, GS11, HTAC19, AT17], namely the need of the context C. To see that this context is necessary, consider the following example graph:

(6.2)

which is of the form $G \,\mathbin{\mathscr{V}}\, (a \otimes b)$ for some G and it is provable in GS. It is impossible however to apply a derivation to G to obtain two disjoint graphs K_A and K_B, such that $K_A \,\mathbin{\mathscr{V}}\, a$ and

---

[11]In the language of proof nets, our splitting lemma says that every tensor is splitting for its empire *and* the proof system is expressive enough to reduce everything else without touching that empire. However, for our graphs, we do not (yet) have a notion of *empire*.

$K_B \,\mathfrak{N}\, b$ are provable, unless we define a suitable context for $K_A \,\mathfrak{N}\, K_B$. Indeed we do have the following context formed from the provable graph $C = g^\perp \,\mathfrak{N}\, f^\perp \,\mathfrak{N}\, (g \otimes f)$.

$$C[\cdot]_{\{f^\perp, f, g\}} = \quad \overset{g^\perp \!-\! f^\perp \!-\! \boxed{} \!-\! f}{\diagdown}{}_{g}$$

 Following the naming convention in the statement of Lemma 6.1, we have $K_A = a^\perp$ and $K_A = b^\perp$ such that $\vdash_{\mathsf{GS}} K_A \,\mathfrak{N}\, a$ and $\vdash_{\mathsf{GS}} K_B \,\mathfrak{N}\, b$ hold, and furthermore G is formed from plugging $K_A \,\mathfrak{N}\, K_B$ insider the above mentioned context formed from C.

From the above ingredients satisfying the condition of the splitting lemma we can always construct a proof. We can apply $\mathsf{s.sw}{\downarrow}$ twice to bring K_A and a together inside a context formed from C, and similarly for K_B and b. After having completed the proofs that consume a and b (by a simple application of $\mathsf{ai}{\downarrow}$) we are left with the provable graph C. The important observation here is that it is impossible to apply a derivation that changes the shape of C until we have completed the derivation that consumes $a^\perp$ and $b^\perp$.

The problem illustrated above with the context C, that cannot be removed until the end of the proof, is a fundamental problem that we have with graphs and that we do not have with formulas. Essentially it is due to the presence of $\mathsf{P_4}$ induced sub-graphs that create indirect dependencies between atoms, such as the indirect dependency between $g^\perp$ and g in the above example.

Generalising splitting to prime graphs. The general idea of a splitting lemma is that, in a provable "sequent-like graph", consisting of a number of disjoint connected components, we can select any of these components as the *principal graph* and apply a derivation to the other components, such that eventually a rule breaking down the principal component can be applied. This allows us to approximate the effect of applying rules in the sequent calculus, even when no sequent calculus exists, as is the case when the principal graph is formed using a prime graph that is not a tensor.

Lemma 6.3 (Splitting Prime). *Let G be a graph and $P \neq \mathfrak{N}$ be a prime graph with $|V_P| = n$, and $M_1, \ldots, M_n$ be non-empty graphs. If $\vdash_{\mathsf{GS}} G \,\mathfrak{N}\, P(\!(M_1, \ldots, M_n)\!)$, then one of the following holds:*

(A) *either there is a context $C[\cdot]_R$ and graphs $K_1, \ldots, K_n$, such that there are derivations*

$$\begin{array}{ccc}
C[P^\perp(\!(K_1, \ldots, K_n)\!)]_R & \varnothing & \varnothing \\
\mathcal{D}_G \,\Big\|\, \mathsf{GS} & \mathcal{D}_i \,\Big\|\, \mathsf{GS} \quad and \quad & \mathcal{D}_C \,\Big\|\, \mathsf{GS} \\
G & K_i \,\mathfrak{N}\, M_i & C
\end{array}$$

for all $i \in \{1, \ldots, n\}$,

(B) *or there is a context $C[\cdot]_R$ and graphs K_X and K_Y, such that there are derivations*

$$\begin{array}{cccc}
C[K_X \,\mathfrak{N}\, K_Y]_R & \varnothing & \varnothing & \varnothing \\
\mathcal{D}_G \,\Big\|\, \mathsf{GS} & \mathcal{D}_X \,\Big\|\, \mathsf{GS} & \mathcal{D}_Y \,\Big\|\, \mathsf{GS} \quad and \quad & \mathcal{D}_C \,\Big\|\, \mathsf{GS} \\
G & K_X \,\mathfrak{N}\, M_i & K_Y \,\mathfrak{N}\, P(\!(M_1, \ldots, M_{i-1}, \varnothing, M_{i+1}, \ldots, M_n)\!) & C
\end{array}$$

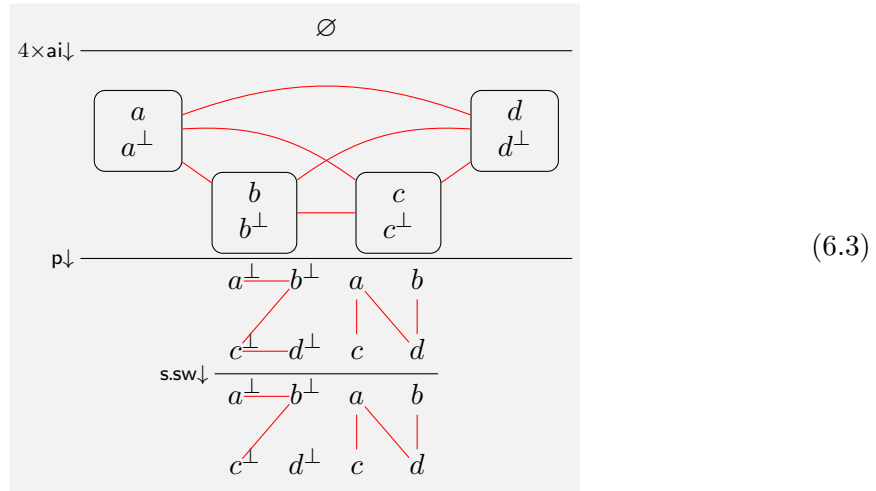*for some $i \in \{1, \ldots, n\}$.*

Remark 6.4. The immediate question to address at this point is why we need two cases, where splitting lemmas in the literature require only one case. Notice that Lemma 6.1 is a special case of Lemma 6.3, since $\otimes$ is a prime graph, as indicated in (3.2) in Section 3.

In the case of the tensor, the two sub-cases of Lemma 6.3 collapse; hence for $\otimes$ the above lemma for prime graphs collapses to the form of Lemma 6.1.

The existence of these two separate cases is a surprising novelty since all proof systems, on which the splitting technique has been applied so far, only employ atoms, binary connectives [Gug07, AT17], modalities [Str03a, GS11], and quantifiers [HTAC19]. Here, for the first time we present splitting for n-ary connectives with $n > 2$.

When the principal graph is formed using a prime graph with at least four vertices, the two cases of Lemma 6.1 are properly distinct. We illustrate first the case (A) of Lemma 6.1, by renormalising the proof (6.1) as an example to explain this idea.

In the conclusion of (6.1) we have the disjoint union of three connected graphs. We can select $\mathsf{P}_4(\!(c, a, d, b)\!)$ (the connected component on the right) as the principal component, and apply Case (A) of Lemma 6.3 to reorganise the derivation in such a way an instance of $\mathsf{p}\!\downarrow$ involving $\mathsf{P}_4(\!(c, a, d, b)\!)$ can be applied (see Example 4.16). This allows us to conclude as in Case (A) applies, since all of four graphs $a^\perp \mathbin{⅋} a$, $b^\perp \mathbin{⅋} b$, $c^\perp \mathbin{⅋} c$ and $d^\perp \mathbin{⅋} d$ are provable. That is, we can obtain a derivation of the form

(6.3)

To see where Case (B) is required in order to destroy a prime graph by removing one of its modules, consider the graph below:

(6.4)

Here Case (A) cannot be applied since when we select $\mathsf{P}_4(\!(c, a^\perp, b^\perp, c^\perp)\!)$ as the principal component and we try to apply $\mathsf{p}\!\downarrow$, then c and $c^\perp$ can no longer communicate. Therefore, we must first move a or b into the structure and apply an $\mathsf{ai}\!\downarrow$, in order to destroy the prime graph in such a way that c and $c^\perp$ can still communicate.

Applying Case (B) to the graph 6.4, following the naming convention in Lemma 6.3, we have graphs $K_X = a$ and $K_Y = b$ such that the following two graphs are provable, as required for splitting.

(6.5)

Then we can reassemble (using $\mathsf{s.sw}{\downarrow}$) the first steps of proof below

Splitting for Atoms. We also need a splitting lemma where the principal graph is just a singleton.

Lemma 6.5 (Atomic Splitting)**.** *Let G be a graph. If $\vdash_{\mathsf{GS}} G \,\invamp\, a$ for an atom a, then there is a context $C[\cdot]_R$ such that there are derivations*

$$
\begin{array}{c} C[a^\perp]_R \\ \mathcal{D}_G \,\|\, \mathsf{GS} \\ G \end{array}
\qquad and \qquad
\begin{array}{c} \varnothing \\ \mathcal{D}_C \,\|\, \mathsf{GS} \\ C \end{array} \quad .
$$

Atomic splitting simply states that for an atom a in the proof, there is a matching $a^\perp$ somewhere else in the proof. Note that there may be more than one $a^\perp$, in which case the lemma will pick out exactly the instance that interacts with a via an $\mathsf{ai}{\downarrow}$ instance in the original proof.

As with splitting for prime graphs the novelty compared to the literature is the presence of the context C.

Context reduction. Observe that the splitting lemma is not applied to a $\invamp$-graph. The $\invamp$ plays in GS a similar role as the comma in the sequent calculus. Splitting lemmas such as Lemma 6.3 apply in a sequent-like context, sometimes called a *shallow context*. Splitting cannot be applied to proofs where the principal graph (that is not a $\invamp$) appears at any depth in the conclusion of the proof. In order to be able to use splitting to eliminate cuts at arbitrary depth, we need to extend splitting from shallow contexts to deep contexts. For this, we need the context reduction lemma, stated below.

Lemma 6.6 (Context reduction)**.** *Let A be a graph and $G[\cdot]_S$ be a context, such that $\vdash_{\mathsf{GS}} G[A]_S$. Then there are a context $C[\cdot]_R$ and a graph K, such that there are derivations*

$$
\begin{array}{c} C[K \,\invamp\, \mathcal{X}]_R \\ \mathcal{D}_G \,\|\, \mathsf{GS} \\ G[\mathcal{X}]_S \end{array}
\qquad and \qquad
\begin{array}{c} \varnothing \\ \mathcal{D}_A \,\|\, \mathsf{GS} \\ K \,\invamp\, A \end{array}
\qquad and \qquad
\begin{array}{c} \varnothing \\ \mathcal{D}_C \,\|\, \mathsf{GS} \\ C \end{array}
$$

for any graph $\mathcal{X}$.

The idea of context reduction is that, if we select any module, called A in Lemma 6.6 above, we can always rewrite its context, say $G[\cdot]_S$, such that it is of the form $K \mathbin{\rotatebox[origin=c]{180}{\&}} [\cdot]_\varnothing$ inside some provable context. The use of $\mathcal{X}$ reflects the fact that, in doing so, you never need to change A; hence A could be replaced by another graph $\mathcal{X}$ and the same transformation can be applied to the context.

Similarly to splitting, the novelty compared to context reduction in the literature is that we must keep track of a provable context. This is required to cope with graphs such as the following:

(6.6)

Suppose that we are interested in the singleton module $a^\perp$ in the above graph, and we wish to apply context reduction to it. When we apply Lemma 6.6, clearly there is only one choice of graph K such that $K \mathbin{\rotatebox[origin=c]{180}{\&}} a^\perp$ is provable, namely $K = a$. Observe, furthermore, that in order to rewrite the rest of the above graph, so it is of the form $C[K \mathbin{\rotatebox[origin=c]{180}{\&}} a^\perp]_R$ such that C is provable, the only choice is to set $C = b^\perp \mathbin{\rotatebox[origin=c]{180}{\&}} c^\perp \mathbin{\rotatebox[origin=c]{180}{\&}} (b \otimes c)$, as done in the following derivation:

(6.7)

which can be completed to a proof, by applying $\mathsf{ai}{\downarrow}$ to the module $a \mathbin{\rotatebox[origin=c]{180}{\&}} a^\perp$ and then $\mathsf{i}{\downarrow}$ to prove the resulting graph C. Examples such as (6.2), (6.4) and (6.7), where each conclusion can only be proven if rules can be applied deep inside a graph instead of to a graph at the top level of the modular decomposition of the conclusion, show that deep inference is necessary for developing a proof theory on graphs allowing $\mathsf{P_4}$ as an induced sub-graph.

For a more substantial example consider the derivation in Figure 2. Assume, we aim to apply splitting to the N-shaped induced subgraph in the conclusion consisting of vertices labelled c, $a^\perp$, $b^\perp$ and $c^\perp$. Clearly, the splitting lemma cannot be directly applied to that induced subgraph, so firstly we reduce the context. The three bottommost inference steps shown in Figure 2 correspond to the steps taken by context reduction that do not touch the N-shaped subgraph but bring the vertices labelled a and b next to our subgraph. At that point Case (B) of splitting for prime graphs can be applied to our N-shaped subgraph, taking into account that, together with the vertices labelled a and b, they form a sequent-like module of the graph at that point in the derivation. At this point, the splitting lemma provides us with the information needed to complete the proof.

An additional reason why Figure 2 is interesting is that it is an example of a proof where the effect of the $\mathsf{p}{\downarrow}$-rule cannot be simulated by using the $\mathsf{i}{\downarrow}$-rule. Therefore, this example emphasises the need for $\mathsf{p}{\downarrow}$ or at least a rule to the effect of $\mathsf{p}{\downarrow}$ in an analytic proof system on graphs.

Proving splitting and context reduction. The detailed proofs of these lemmas can be found in Appendix A. Here we present an outline and the basic ideas. As standard, we

Figure 2: A proof used to illustrate context reduction and splitting. It also shows a non-trivial use of the $\mathsf{p}\!\downarrow$-rule that cannot be achieved using the $\mathsf{i}\!\downarrow$-rule.

proceed by exhausting all possible permutations of rules where the main points of interest are where the rules change the principal connective in some way. The novelty compared to results in the literature on deep inference is that, due to the important role of contexts we prove splitting and context reduction together by mutual induction, using the size $\|G\|$ of a graph (see Definition 5.13) as induction measure.

During the proof of splitting and context reduction we must handle cases involving the $\mathsf{p}\!\downarrow$ rule. Observe that when the $\mathsf{p}\!\downarrow$ rule is applied we obtain at least four graphs connected by tensors. Since we will need to handle frequently such cases, we find it convenient to state splitting for such multi-tensors as a lemma. In what follows we generalise the Lemma 6.1 to the n-ary tensor for any $n \geq 2$ (or *multi-tensor*), which is clearly a consequence of splitting for tensor (which, in turn, we have discussed is a special case of splitting for prime graphs).

Lemma 6.7 (Splitting Multi-tensor). *Let G be a graph, and $A_1, \ldots, A_n$ be non-empty graphs. If $\vdash_{\mathsf{GS}} G \bindnasrepma (A_1 \otimes \cdots \otimes A_n)$, then there is a context $C[\cdot]_R$ and graphs $K_1, \ldots, K_n$, such that there are derivations*

$$\begin{array}{c} C[K_1 \bindnasrepma \cdots \bindnasrepma K_n]_R \\ \mathcal{D}_G \,\big\|\, \mathsf{GS} \\ G \end{array} \quad , \quad \begin{array}{c} \varnothing \\ \mathcal{D}_i \,\big\|\, \mathsf{GS} \\ K_i \bindnasrepma A_i \end{array} \quad and \quad \begin{array}{c} \varnothing \\ \mathcal{D}_C \,\big\|\, \mathsf{GS} \\ C \end{array}$$

for all $i \in \{1, \ldots, n\}$.

Figure 3: Complete roadmap of splitting and context reduction lemmas proofs. Double head arrows make use of Lemma 5.1.

All lemmas of this section are proved simultaneously, by case analysis and an inductive argument. Before we go into the details of the case analysis, let us draw the attention to Figure 3, which shows in more detail the dependencies of the proofs of all lemmas in this section. The main lemma is Lemma 6.3. Then Lemma 6.1 is just a special case of Lemma 6.3, and Lemma 6.7 follows immediately by iterating Lemma 6.1. To ensure that there is no circular reasoning in the other dependencies, we argue that whenever the proof of one lemma refers to another then this happens with respect to a strictly smaller graph (according to Definition 5.13). Let us now look more closely at the case analysis for Lemma 6.3.

Case Analysis for the proof of Lemma 6.3 (Splitting Prime). There are five cases to consider which we enumerate below.

(a) The last rule in the given proof acts inside G or any M_i with $i \in \{1, \ldots, n\}$. This case is relatively simple, since the structure of the conclusion does not change.

(b) The last rule in the given proof is a $\mathsf{s.sw}{\downarrow}$ such that another graph, external to the principal prime graph, moves inside the principal prime graph, i.e., $G = G' \,\invamp\, G''$ with $G' \neq \varnothing$ and $\mathcal{D}$ is of shape

$$\mathsf{s.sw}{\downarrow} \frac{\begin{array}{c} \varnothing \\ \mathcal{D}'' \,\Big\|\, \mathsf{GS} \\ G'' \,\invamp\, P(\!|M_1, \ldots, M_n|\!)[G']_{R_P} \end{array}}{G'' \,\invamp\, G' \,\invamp\, P(\!|M_1, \ldots, M_n|\!)}$$

This can result in several scenarios, that are named (b.I), (b.II), and (b.III) in the appendix, and that are not all immediately obvious.

(c) The last rule in the given proof is a $\mathsf{s.sw}\!\downarrow$ that moves the whole prime graph inside the rest of the graph. In this case $\mathcal{D}$ is of shape

$$\mathsf{s.sw}\!\downarrow \frac{\genfrac{}{}{0pt}{}{\varnothing}{\mathcal{D}'\,\|\,\mathsf{GS}}\quad G[P(\!|M_1,\ldots,M_n|\!)]_S}{G \,⅋\, P(\!|M_1,\ldots,M_n|\!)}$$

We will explain that this case appeals to context reduction.

(d) The last rule in the given proof is a $\mathsf{p}\!\downarrow$ directly applied to the principal connective, i.e., $G = G' \,⅋\, P^{\perp}(\!|N_1,\ldots,N_n|\!)$ and $\mathcal{D}$ is of shape

$$\mathsf{p}\!\downarrow \frac{\genfrac{}{}{0pt}{}{\varnothing}{\mathcal{D}'\,\|\,\mathsf{GS}}\quad G'' \,⅋\, ((N_1 \,⅋\, M_1) \otimes \cdots \otimes (N_n \,⅋\, M_n))}{G'' \,⅋\, P^{\perp}(\!|N_1,\ldots,N_n|\!) \,⅋\, P(\!|M_1,\ldots,M_n|\!)} \; .$$

In this case we have to refer to the multi-tensor splitting lemma.

(e) The last rule is a $\mathsf{p}\!\downarrow$ such that not all components of the principal connective are non-empty, and we have $G = G'' \,⅋\, Q(\!|N_1,\ldots,N_k|\!)$ where $N_1,\ldots,N_k$ are non-empty graphs, Q is a prime graph with $|V_Q| > |V_P|$ and such that w.l.o.g. $P(\!|M_1,\ldots,M_n|\!) = Q^{\perp}(\!|\varnothing, L_2, \ldots L_k|\!)$ for some (possibly empty) graphs $L_2, \ldots L_k$, and $\mathcal{D}$ is of shape

$$\mathsf{p}\!\downarrow \frac{\genfrac{}{}{0pt}{}{\varnothing}{\mathcal{D}'\,\|\,\mathsf{GS}}\quad G'' \,⅋\, (N_1 \otimes (N_2 \,⅋\, L_2) \otimes \ldots \otimes (N_k \,⅋\, L_k))}{G'' \,⅋\, Q(\!|N_1,\ldots,N_k|\!) \,⅋\, P(\!|M_1,\ldots,M_n|\!)}$$

In this case it is important to ensure the side-conditions on $\mathsf{p}\!\downarrow$ are respected.

All technical details of this case analysis are in Appendix A. In the remainder of this section, we give some informal explanation why the cases in the proof of Lemma 6.3 can become complex. For this, consider the following example.

(6.8)

One possible way to prove the above graph is to first apply the $\mathsf{s.sw}\!\downarrow$-rule to move the vertex labelled e so that it is attached by an edge only to h. This way we obtain exactly the graph in the conclusion of Figure 2, therefore we know already how to complete the proof via that strategy.

Now observe that the above graph (6.8) is of the form $G \,⅋\, ((a \,⅋\, b) \otimes (h \,⅋\, d))$. Thus it is in a form to which we could apply tensor splitting (Lemma 6.1) to this $\otimes$-operator as the principal graph. This forces the proof to be renormalised such that firstly e moves next to

$e^{\perp}$ as shown in the derivation below.

Observe that in the topmost graph in the derivation above we have broken the graph on the left into two disjoint graphs. The first is provable when composed with $a \,\bindnasrepma\, b$, while the second is provable when composed with $d \,\bindnasrepma\, h$.

The choices between the different rules to first apply to graph (6.8), where s.sw↓ is applied to e in order to either create a larger prime graph on the right (leading to the proof in Figure 2) or to break up the structure on the left as shown above, lead to quite different proofs. Moving from the former to the latter proof is discussed in Case (b.III) in the proof of Lemma 6.3, which is in fact the most involved sub-case.

Case (b.II) in the proof of Lemma 6.3 is not immediately obvious but is nonetheless usually present in standard splitting proofs on formulas, where similar cases are induced by associativity. A typical example is given by the following two isomorphic graphs, that are composed differently:

This isomorphism is used in the proof below in order to enable the s.sw↓-rule:

$$(6.9)$$

We can apply splitting to the above graph in various ways. One possibility is to apply splitting to the tensor on the right as shown in the conclusion of the above proof. For that we appeal to Case (b.II) in the proof of Lemma 6.3, which firstly observes that there is a

proof of $e^\perp \,\bindnasrepma\, e$ and a proof of the following induced sub-graph.

$$
\begin{array}{ccc}
a^\perp \quad f^\perp \quad c^\perp & & \\
\big| \quad\quad\quad \big| & f & \boxed{\begin{array}{c} a \\ b \end{array}} \!-\! \boxed{\begin{array}{c} c \\ d \end{array}}
\end{array}
\tag{6.10}
$$

Case (b.III) is then used to renormalise the proof of (6.10) such that the P_5 on the left is split into two parts, from which we can reassemble a proof of the graph in the conclusion of (6.9) which satisfies the conditions of splitting for the tensor that we selected. In particular, if $A = e \otimes (a \,\bindnasrepma\, b)$ and $B = c \,\bindnasrepma\, d$ in Lemma 6.1, then we obtain $K_A = (a^\perp \otimes b^\perp) \,\bindnasrepma\, e^\perp$ and $K_B = c^\perp \,\bindnasrepma\, d^\perp$, as required (the context is empty for this example).

Beyond the above two illustrative examples of non-trivial normalisation steps, observe that the deepest nesting in the proof of splitting for prime graphs is inside Case (b.III), which has two more levels of nested case analysis to cover all ways in which we can handle situations where the principal graph is turned into a larger prime graph using the s.sw↓ rule.

7. Elimination of the Up-Fragment

We do now have all ingredients to prove the cut elimination result, i.e., the rule i↑ is admissible for GS (Theorem 5.4). Recall that this is equivalent to the admissibility of the up-rules ai↑, s.sw↑, and p↑ (Theorem 5.5), and that this entails the transitivity of the consequence relation (Lemma 4.15) and consistency of the proof system (Corollary 5.11).

In this section we are going to show how splitting and context reduction are employed to prove this result. The procedure is similar to ordinary deep inference systems (see, e.g., [Str03a, GS11, HTAC19, CGS11]).

Theorem 7.1. *The rule* ai↑ *is admissible for* GS.

Proof. Assume we have a proof of $G[a \otimes a^\perp]_S$. By Lemma 6.6 we have a graph L and a context $C_1[\cdot]_{R_1}$, such that there are derivations

$$
\begin{array}{ccccc}
\varnothing & & \varnothing & & C_1[L \,\bindnasrepma\, \mathcal{X}]_{R_1} \\
\mathcal{D}_1 \,\|\, \mathsf{GS} & , & \mathcal{D}_2 \,\|\, \mathsf{GS} & \text{and} & \mathcal{D}_3 \,\|\, \mathsf{GS} \\
C_1 & & L \,\bindnasrepma\, (a \otimes a^\perp) & & G[\mathcal{X}]_S
\end{array}
$$

for any graph $\mathcal{X}$. We apply Lemma 6.1 to $L \,\bindnasrepma\, (a \otimes a^\perp)$ and get K_a and $K_{a^\perp}$ and a context $C_2[\cdot]_{R_2}$ such that

$$
\begin{array}{ccccccc}
C_2[K_a \,\bindnasrepma\, K_{a^\perp}]_{R_2} & & \varnothing & & \varnothing & & \varnothing \\
\mathcal{D}_4 \,\|\, \mathsf{GS} & , & \mathcal{D}_6 \,\|\, \mathsf{GS} & , & \mathcal{D}_7 \,\|\, \mathsf{GS} & \text{and} & \mathcal{D}_5 \,\|\, \mathsf{GS} \\
L & & K_a \,\bindnasrepma\, a & & K_{a^\perp} \,\bindnasrepma\, a^\perp & & C_2
\end{array}
$$

Applying Lemma 6.5 to $K_a \,\bindnasrepma\, a$ and $K_{a^\perp} \,\bindnasrepma\, a^\perp$ gives us $C_3[\cdot]_{R_3}$ and $C_4[\cdot]_{R_4}$ such that

$$
\begin{array}{ccccccc}
C_3[a^\perp]_{R_3} & & \varnothing & & C_4[a]_{R_4} & & \varnothing \\
\mathcal{D}_8 \,\|\, \mathsf{GS} & , & \mathcal{D}_9 \,\|\, \mathsf{GS} & , & \mathcal{D}_{10} \,\|\, \mathsf{GS} & \text{and} & \mathcal{D}_{11} \,\|\, \mathsf{GS} \\
K_a & & C_3 & & K_{a^\perp} & & C_4
\end{array}
$$

We can now give the following derivation

$$
C_1 \left|\; C_2 \left[\; C_3 \left[\; C_4 \left[\mathsf{ai}{\downarrow} \frac{\varnothing}{a^\perp \,\math>\!\!\!\!\gamma\, a} \right] \right] \right] \right]
$$

that proves $G = G[\varnothing]_S$ in GS.   □

Theorem 7.2. *The rule* s.sw↑ *is admissible for* GS.

Proof. Assume we have a proof of $G[B \otimes A]_S$ in GS. By Lemma 6.6 we have a graph L and a context $C_1[\cdot]_{R_1}$, such that there are derivations

$$
\begin{array}{c} \varnothing \\ \mathcal{D}_1 \,\|\, \mathsf{GS} \\ C_1 \end{array} \;,\quad
\begin{array}{c} \varnothing \\ \mathcal{D}_2 \,\|\, \mathsf{GS} \\ L \,\math>\!\!\!\!\gamma\, (B \otimes A) \end{array} \quad \text{and} \quad
\begin{array}{c} C_1[L \,\math>\!\!\!\!\gamma\, \mathcal{X}]_{R_1} \\ \mathcal{D}_3 \,\|\, \mathsf{GS} \\ G[\mathcal{X}]_S \end{array}
$$

for any graph $\mathcal{X}$. We apply Lemma 6.1 to $L \,\math>\!\!\!\!\gamma\, (B \otimes A)$ and get K_B and K_A and a context $C_2[\cdot]_{R_2}$ such that

$$
\begin{array}{c} C_2[K_B \,\math>\!\!\!\!\gamma\, K_A]_{R_2} \\ \mathcal{D}_4 \,\|\, \mathsf{GS} \\ L \end{array} \;,\quad
\begin{array}{c} \varnothing \\ \mathcal{D}_6 \,\|\, \mathsf{GS} \\ K_B \,\math>\!\!\!\!\gamma\, B \end{array} \;,\quad
\begin{array}{c} \varnothing \\ \mathcal{D}_7 \,\|\, \mathsf{GS} \\ K_A \,\math>\!\!\!\!\gamma\, A \end{array} \quad \text{and} \quad
\begin{array}{c} \varnothing \\ \mathcal{D}_5 \,\|\, \mathsf{GS} \\ C_2 \end{array} \;.
$$

We can now give a proof of $G[B[A]_T]_S$ as follows:

for any $\varnothing \subseteq T \subset |V_B|$. $\qquad\square$

Theorem 7.3. *The rule* p↑ *is admissible for* GS.

Proof. This proof is slightly different from the previous two and from what usually happens in a deep inference setting. In particular, we need to invoke an induction on the "size of the cut formula", which in our case is the size of the module $P(\!|M_1, \ldots, M_n|\!) \otimes P^\perp(\!|N_1, \ldots, N_n|\!)$ in the premise of the rule. In other cut elimination proofs in deep inference, there is no need for such an induction, as it is outsourced to the splitting lemma.

Now, assume we have a proof of $G[P(\!|M_1, \ldots, M_n|\!) \otimes P^\perp(\!|N_1, \ldots, N_n|\!)]_S$ with $M_1, \ldots, M_n$ non-empty graphs. We apply Lemma 6.6 and get a graph L and a context $C_1[\cdot]_{R_1}$, such that there are derivations

$$
\begin{array}{ccc}
\varnothing & \varnothing & C_1[L \,\mathbin{\text{⅋}}\, \mathcal{X}]_{R_1} \\
\mathcal{D}_1 \,\|\, \mathsf{GS} \quad, & \mathcal{D}_2 \,\|\, \mathsf{GS} & \text{and} \qquad \mathcal{D}_3 \,\|\, \mathsf{GS} \\
C_1 & L \,\mathbin{\text{⅋}}\, (P(\!|M_1, \ldots, M_n|\!) \otimes P^\perp(\!|N_1, \ldots, N_n|\!)) & G[\mathcal{X}]_S
\end{array}
$$

for any graph $\mathcal{X}$.

- First, consider the case where $P^\perp(\!|N_1, \ldots, N_n|\!) \neq \varnothing$. We can apply Lemma 6.1 to $L \mathbin{\text{⅋}} (P(\!|M_1, \ldots, M_n|\!) \otimes P^\perp(\!|N_1, \ldots, N_n|\!))$ and get graphs L_P and $L_{P^\perp}$ and a context $C_2[\cdot]_{R_2}$ such that

$$
\begin{array}{cccc}
C_2[L_P \,\mathbin{\text{⅋}}\, L_{P^\perp}]_{R_2} & \varnothing & \varnothing & \varnothing \\
\mathcal{D}_4 \,\|\, \mathsf{GS} \quad, & \mathcal{D}_5 \,\|\, \mathsf{GS} \quad, & \mathcal{D}_6 \,\|\, \mathsf{GS} & \text{and} \quad \mathcal{D}_7 \,\|\, \mathsf{GS} \ . \\
L & L_P \,\mathbin{\text{⅋}}\, P(\!|M_1, \ldots, M_n|\!) & L_{P^\perp} \,\mathbin{\text{⅋}}\, P^\perp(\!|N_1, \ldots, N_n|\!) & C_2
\end{array}
$$

Applying Lemma 6.3 to $L_P \mathbin{\text{⅋}} P(\!|M_1, \ldots, M_n|\!)$ gives us two different cases.

(A) There are graphs $K_1, \ldots, K_n$ and a context $C_3[\cdot]_{R_3}$ such that

$$
\begin{array}{ccc}
\begin{array}{c} C_3[P^\perp(\!|K_1, \ldots, K_n|\!)]_{R_3} \\ \mathcal{D}_8 \,\|\, \mathsf{GS} \\ L_P \end{array} &
\begin{array}{c} \varnothing \\ \mathcal{D}'_i \,\|\, \mathsf{GS} \\ K_i \,\mathbin{\text{⅋}}\, M_i \end{array} &
\begin{array}{c} \varnothing \\ \mathcal{D}_9 \,\|\, \mathsf{GS} \\ C_3 \end{array}
\end{array},
$$

for all $i \in \{1, \ldots, n\}$. Then, our derivation of $G[(M_1 \otimes N_1) \mathbin{\text{⅋}} \cdots \mathbin{\text{⅋}} (M_n \otimes N_n)]_S$ is constructed as follows

(B) There are graphs H_X and H_Y and a context $C_4[\cdot]_{R_4}$, such that

$$
\begin{array}{cccc}
\begin{array}{c} C_4[H_X \mathbin{\text{⅋}} H_Y]_{R_4} \\ \mathcal{D}_{10} \,\|\, \mathsf{GS} \\ L_P \end{array} &
\begin{array}{c} \varnothing \\ \mathcal{D}_X \,\|\, \mathsf{GS} \\ H_X \mathbin{\text{⅋}} M_j \end{array} &
\begin{array}{c} \varnothing \\ \mathcal{D}_Y \,\|\, \mathsf{GS} \\ H_Y \mathbin{\text{⅋}} P(\!|M_1, \ldots, M_{i-1}, \varnothing, M_{i+1}, \ldots M_n|\!) \end{array} & \text{and} \quad
\begin{array}{c} \varnothing \\ \mathcal{D}_{11} \,\|\, \mathsf{GS} \\ C_4 \end{array}
\end{array}
$$

for some $i \in \{1, \ldots, n\}$.

We apply context reduction (Lemma 6.6) to the derivation $\mathcal{D}_6$ with conclusion

$$
L_{P^\perp} \mathbin{\text{⅋}} P(\!|N_1, \ldots, N_{j-1}, N_j, N_{j+1}, \ldots, N_n|\!)
$$

to obtain a context C_5 and a graph K_{N_j} such that

$$
\begin{array}{ccc}
\begin{array}{c} C_5[K_{N_j} \mathbin{\text{⅋}} \mathcal{Y}]_{R_5} \\ \mathcal{D}^{\mathcal{Y}}_{12} \,\|\, \mathsf{GS} \\ L_{P^\perp} \mathbin{\text{⅋}} P^\perp(\!|N_1, \ldots, N_{j-1}, \mathcal{Y}, N_{j+1}, \ldots, N_n|\!) \end{array} &
\begin{array}{c} \varnothing \\ \mathcal{D}_{13} \,\|\, \mathsf{GS} \\ N_j \mathbin{\text{⅋}} K_{N_j} \end{array} & \text{and} \quad
\begin{array}{c} \varnothing \\ \mathcal{D}_{14} \,\|\, \mathsf{GS} \\ C_5 \end{array}
\end{array}
$$

For every graph $\mathcal{Y}$. Then, our proof of $G[(M_1 \otimes N_1) \mathbin{\text{⅋}} \cdots \mathbin{\text{⅋}} (M_n \otimes N_n)]_S$ is constructed as shown in Figure 4.

Figure 4: Derivation for case (B) in the proof of Theorem 7.3

Observe that the derivation $\mathcal{D}^*$ in Figure 4 is a derivation in $\mathsf{SGS{\uparrow}}$. However, all instances of $\mathsf{ai{\uparrow}}$ and $\mathsf{s.sw{\uparrow}}$ in $\mathcal{D}^*$ can be eliminated using using Theorems 7.1 and 7.2, while, in order to eliminate the instances of $\mathsf{p{\uparrow}}$, we invoke the induction hypothesis since the principal modules in the $\mathsf{p{\uparrow}}$ instances in $\mathcal{D}^*$ have strictly smaller size than the one we started with.

- Now assume $P^\perp(\!|N_1,\ldots,N_n|\!) = \varnothing$. Then by context reduction we already have a derivation $\mathcal{D}_5 = \mathcal{D}_2$ of $L \,\mathfrak{N}\, (P(\!|M_1,\ldots,M_n|\!) \otimes P^\perp(\!|N_1,\ldots,N_n|\!)) \simeq L_P \,\mathfrak{N}\, P(\!|M_1,\ldots,M_n|\!)$, to which we apply Lemma 6.3 as above. Then case (A) is identical to the one above, and case (B) is simplified to

Alternative formulations of the rules of GS. Further admissibility results follow immediately without having to pass via splitting and context reduction, now that we have established cut elimination. We can in fact considerably strengthen Theorem 5.5, as follows.

Corollary 7.4. *Let r be an inference rule. If for every instance $\mathsf{r}\dfrac{A}{B}$ of that rule we have $\vdash_{\mathsf{GS}} A \multimap B$, then r is admissible for GS.*

Proof. Assume we have $\vdash_{\mathsf{GS}} A$. By Corollary 5.7 we have a derivation in SGS from A to B. Hence, we have $\vdash_{\mathsf{SGS}} B$, and therefore by Corollary 5.6 also $\vdash_{\mathsf{GS}} B$. $\qquad\square$

Now consider the following two inference rules:

$$\mathsf{g{\downarrow}}\frac{(M_1 \,\mathfrak{N}\, N_1) \otimes \cdots \otimes (M_n \,\mathfrak{N}\, N_n)}{G(\!|M_1,\ldots,M_n|\!) \,\mathfrak{N}\, G^\perp(\!|N_1,\ldots,N_n|\!)} \qquad \mathsf{g{\uparrow}}\frac{G(\!|M_1,\ldots,M_n|\!) \otimes G^\perp(\!|N_1,\ldots,N_n|\!)}{(M_1 \otimes N_1) \,\mathfrak{N}\, \cdots \,\mathfrak{N}\, (M_n \otimes N_n)} \qquad (7.1)$$

Their shape is similar to $\mathsf{p{\downarrow}}$ and $\mathsf{p{\uparrow}}$, but there are no side conditions, i.e., G can be any graph, and there are no emptiness or non-emptiness conditions for M_i or N_i.

Corollary 7.5. *The two rules $\mathsf{g{\downarrow}}$ and $\mathsf{g{\uparrow}}$ are admissible for GS.*

Proof. We are going to show that for all graphs G with $|V_G| = n$ and graphs $M_1, N_1, \ldots, M_n, N_n$ we have that

$$\vdash_{\mathsf{GS}} (M_1^\perp \otimes N_1^\perp) \,\mathfrak{N}\, \cdots \,\mathfrak{N}\, (M_n^\perp \otimes N_n^\perp) \,\mathfrak{N}\, G(\!|M_1,\ldots,M_n|\!) \,\mathfrak{N}\, G^\perp(\!|N_1,\ldots,N_n|\!) \quad . \qquad (7.2)$$

Then this corollary follows via Corollaries 5.8 and 7.4. To prove (7.2), we use the following derivation

$$
n\times\mathsf{i}\downarrow \frac{\varnothing}{((M_1^\perp \otimes N_1^\perp)\,\mathbin{⅋}\, M_1\,\mathbin{⅋}\, N_1)\otimes\cdots\otimes((M_n^\perp \otimes N_n^\perp)\,\mathbin{⅋}\, M_n\,\mathbin{⅋}\, N_n)}
$$

$$
\mathcal{D}\,\Big\|\,\mathsf{GS}
$$

$$
n\times\mathsf{s.sw}\downarrow\frac{G(\!|(M_1^\perp\otimes N_1^\perp)\,\mathbin{⅋}\, M_1,\ldots,(M_n^\perp\otimes N_n^\perp)\,\mathbin{⅋}\, M_n|\!)\,\mathbin{⅋}\, G^\perp(\!|N_1,\ldots,N_n|\!)}{(M_1^\perp\otimes N_1^\perp)\,\mathbin{⅋}\cdots\mathbin{⅋}\,(M_n^\perp\otimes N_n^\perp)\,\mathbin{⅋}\, G(\!|M_1,\ldots,M_n|\!)\,\mathbin{⅋}\, G^\perp(\!|N_1,\ldots,N_n|\!)}
$$

where $\mathcal{D}$ exists by Lemma 5.1. To see this, observe that for each $i\in\{1,\ldots,n\}$, there are two cases: either $(M_i^\perp\otimes N_i^\perp)\,\mathbin{⅋}\, M_i\neq\varnothing$ or $(M_i^\perp\otimes N_i^\perp)\,\mathbin{⅋}\, M_i=\varnothing$. In the second case we also have $N_i^\perp=\varnothing$, and therefore also $N_i=\varnothing$. Hence, the condition for applying Lemma 5.1 is fulfilled. □

The above corollary shows that a variant of GS where we use $\mathsf{g}\downarrow$ instead of $\mathsf{p}\downarrow$ is equivalent to the system GS using only $\mathsf{p}\downarrow$, as it is defined in Figure 1. Indeed, as noted previously in Section 4, a weaker constraint on the $\mathsf{p}\downarrow$ and $\mathsf{p}\uparrow$ was used in the conference version of this paper [AHS20], which did not insist that all of the components of one graph are non-empty when applying these rules. Thus we have tightened the proof search space of GS, but have not changed the expressive power of the logic.

The reason why it is not immediately obvious whether the stronger side conditions in this version of paper can be used, is that Corollary 7.5 is a proper admissibility result rather than a derivability result. As shown in Lemma 5.1, if we drop the condition in $\mathsf{p}\downarrow$ that P is prime then the rule for general graphs is derivable. However, the condition that all components of one graph are non-empty is more subtle, since the resulting variant of $\mathsf{p}\downarrow$ without that side-condition is not derivable using the rules of GS with the side-conditions fixed in Figure 1. For a counterexample to the derivability of the unconstrained $\mathsf{g}\downarrow$ rule in GS, consider the proof on the left below and observe that the instance of $\mathsf{g}\downarrow$ cannot be derived using the rules of GS.

(7.3)

However, observe that there is a proof of the same conclusion on the right above, using the rules of GS with the correct side conditions.

The reason why we use the rules p↓ and p↑ instead of the rules g↓ and g↑ is to have more control over the structure of a derivation, such that we are able to prove cut elimination. In fact, one can say that the main purpose of using the modular decomposition of graphs into prime graphs (instead of arbitrary quotient graphs) is to be able to prove the splitting and context reduction lemmas of Section 6.

8. ANALYTICITY AND CONSERVATIVITY

In sequent calculi, analyticity is often associated to the subformula property, demanding that each subformula in the conclusion of a rule also occurs in its premise. This, in turn, entails that every formula that occurs in a proof is a subformula of the conclusion of the proof.

This notion of analyticity cannot be applied to deep inference, where there is no subformula property, and where in general, analyticity is associated to the possibility of eliminating the up-rules from the system [BG16], such that at any time in the proof search only a finite number of rules can be applied.

In this sense, the rule g↓ discussed above cannot be considered analytic. The absence of constraints on the non-emptiness of the modules in the conclusion allows us to apply the rule in infinitely many ways. By means of example, any application of the p↓-rule to a prime graph P_4 could be considered as an instance of g↓ on any prime graph, since every prime graph contains an induced P_4.

However, the step from formulas/cographs to general graphs allows us to formulate a notion of analyticity for GS, that is more similar to the well-known notion of analyticity for sequent proofs than what is usually present for deep inference systems. For this we introduce the notions of *connector* and *subconnector*, which are for our notion of analyticity what formulas and subformulas are for the standard notion.

Definition 8.1. A prime graph P is a **connector** of a graph G if P occurs in the modular decomposition of G. A **subconnector** of G is a prime graph that is an induced subgraph of a connector of G.

Example 8.2. The graph from Example 3.10 has three occurrences of the connector P_4, one occurrence of the connector $\otimes$ and one occurrence of the connector $⅋$. Note that $\otimes$ is a subconnector of any graph containing at least one edge, and $⅋$ is a subconnector of any graph containing two distinct vertices u and v that are not connected by an edge in G. The graph P_4 is a subconnector of any graph that has a prime connector P with $|V_P| \geq 4$.

Intuitively, if we consider prime graphs as generalised connectives (see Section 9), a connector of G is an occurrence of such a connective in the modular decomposition tree of G. With the help of connectors and subconnectors, we can now formulate our analyticity result.

Theorem 8.3 (Analyticity)**.** *Let G be a graph that is provable in $\mathsf{GS} \cup \{\mathsf{s.sw}\uparrow\}$. Then G admits a derivation $\tilde{\mathcal{D}}$ in $\mathsf{GS} \cup \{\mathsf{s.sw}\uparrow\}$ such that every connector of a graph occurring in $\tilde{\mathcal{D}}$ is a subconnector of G.*

Proof. Let G be given, and let $\mathcal{D}$ be a derivation of G in $\mathsf{GS} \cup \{\mathsf{s.sw}\uparrow\}$. We are now going to transform $\mathcal{D}$ into a derivation $\tilde{\mathcal{D}}$ that does not contain any occurrences of a connector that is not a subconnector of G. Let P be such a connector occuring in $\mathcal{D}$. While going up in the derivation $\mathcal{D}$, only an instance of $\mathsf{s.sw}\downarrow$ can introduce P as connector, i.e., the premise of the rule instance contains the prime graph P as connector, but the conclusion does not. This

$$P\left(\begin{matrix} M_1 \\ \mathcal{D}_1 \| \\ M_1' \end{matrix}, \ldots, \begin{matrix} M_n \\ \mathcal{D}_n \| \\ M_n' \end{matrix}\right) \simeq \cfrac{P\left(M_1, \ldots, M_{i-1}, \begin{matrix} M_1 \\ \mathcal{D}_1 \| \\ M_1' \end{matrix}, M_{i+1}, \ldots, M_n\right)}{P\left(\begin{matrix} M_i \\ \mathcal{D}_i \| \\ M_i' \end{matrix}, \ldots, \begin{matrix} M_{i-1} \\ \mathcal{D}_{i-1} \| \\ M_{i-1}' \end{matrix}, M', \begin{matrix} M_{i+1} \\ \mathcal{D}_{i+1} \| \\ M_{i+1}' \end{matrix}, \ldots, \begin{matrix} M_n \\ \mathcal{D}_n \| \\ M_n' \end{matrix}\right)}$$

$$\text{s.sw}\downarrow \cfrac{C\left[\begin{matrix} G \\ \mathcal{D} \| \\ G' \end{matrix}\right]_R}{C[\varnothing]_R \,\mathcal{P}\, G'} \simeq \text{s.sw}\downarrow \cfrac{\cfrac{C[G]_R}{G}}{C[\varnothing]_R \,\mathcal{P}\, \begin{matrix} \mathcal{D} \| \\ G' \end{matrix}}$$

$$\text{s.sw}\downarrow \cfrac{\cfrac{C[G \,\mathcal{P}\, H]_R}{\text{s.sw}\downarrow \cfrac{C[G]_R}{C[\varnothing]_R \,\mathcal{P}\, G}} \,\mathcal{P}\, H}{} \simeq \text{s.sw}\downarrow \cfrac{C[G \,\mathcal{P}\, H]_R}{C[\varnothing]_R \,\mathcal{P}\, G \,\mathcal{P}\, H}$$

$$\text{p}\downarrow \cfrac{(M_1 \,\mathcal{P}\, N_1) \otimes \cdots \otimes (M_{i-1} \,\mathcal{P}\, N_{i-1}) \otimes \left(\begin{matrix} M_i \\ \mathcal{D}_1 \| \\ M_i' \end{matrix} \,\mathcal{P}\, \begin{matrix} N_i \\ \mathcal{D}_2 \| \\ N_i' \end{matrix}\right) \otimes (M_{i+1} \,\mathcal{P}\, N_{i+1}) \otimes \cdots \otimes (M_n \,\mathcal{P}\, N_n)}{P(M_1, \ldots, M_n) \,\mathcal{P}\, P^\perp(N_1, \ldots, N_n)}$$

$$\simeq \text{p}\downarrow \cfrac{(M_1 \,\mathcal{P}\, N_1) \otimes \cdots \otimes (M_n \,\mathcal{P}\, N_n)}{P\left(M_1, \ldots, M_{i-1}, \begin{matrix} M_i \\ \mathcal{D}_1 \| \\ M_i' \end{matrix}, M_{i+1} M_n\right) \,\mathcal{P}\, P^\perp\left(N_1, \ldots, N_{i-1}, \begin{matrix} N_i \\ \mathcal{D}_2 \| \\ N_i' \end{matrix}, N_{i+1}, \ldots, N_n\right)}$$

Figure 5: Some of the rule permutations captured by the open deduction syntax.

happens by moving a module M_i inside a context $C\big[P(M_1, \ldots, M_{i-1}, \varnothing, M_{i+1}, \ldots, M_n)\big]_R$ where $M_1, \ldots, M_{i-1}, M_{i+1}, \ldots, M_n$ are non-empty. As the premise of the derivation $\mathcal{D}$ is empty, this occurrence of P must be destroyed eventually, i.e., $\mathcal{D}$ contains a rule instance whose conclusion contains P as connector, but the premise does not. This can be done via an instance of s.sw↑, p↓, or ai↓. Below, we will list all cases how this can happen, and show how the derivation can be rewritten without introducing and destroying this occurrence of P. For this, we proceed by induction on the multiset $\{P_1, \ldots, P_n\}$, where $P_1, \ldots, P_n$ are the occurrences of connectors in $\mathcal{D}$ that are not subconnectors of G, and where we use the multiset ordering: $Q < P$ iff $|V_Q| < |V_P|$.

In the discussion of the cases below, we make use of the open deduction notation of derivations (as introduced in Section 4) and the possibility of rule permutations, which are shown in Figure 5. More precisely, we assume we have a connector P introduced by an instance of s.sw↓ as described above, and because of the rule permutations, we can without loss of generality assume that the rule that destroys this connector P is directly above, i.e., we have the following three cases:

(1) An instance of s.sw↑ destroys the connector P by extracting a module M_j from $P(M_1, \ldots, M_n)$. There are two cases to consider: $j = i$ and $j \neq i$.
   (a) If $j = i$, then we can without loss of generality assume that $i = j = 1$, and with the rule permutations of Figure 5, we can move the instances of s.sw↓ and s.sw↑ close

to each other, such that we have one of the following two subderivations:

$$
\text{s.sw}\uparrow \frac{M_1 \otimes C_1\Big[C_2\big[P(\!|\varnothing, M_2, \ldots, M_n|\!)\big]_{R_2}\Big]_{R_1}}{C_1\left[\text{s.sw}\downarrow \dfrac{C_2\big[P(\!|M_1, M_2, \ldots, M_n|\!)\big]_{R_2}}{M_1 \invamp C_2\big[P(\!|\varnothing, M_2, \ldots, M_n|\!)\big]_{R_2}}\right]_{R_1}}
$$

or

$$
\text{s.sw}\downarrow \frac{C_1\left[\text{s.sw}\uparrow \dfrac{M_1 \otimes C_2\big[P(\!|\varnothing, M_2, \ldots, M_n|\!)\big]_{R_2}}{C_2\big[P(\!|M_1, M_2, \ldots, M_n|\!)\big]_{R_2}}\right]_{R_1}}{M_1 \invamp C_1\Big[C_2\big[P(\!|\varnothing, M_2, \ldots, M_n|\!)\big]_{R_2}\Big]_{R_1}}
$$

where the graphs $M_1, \ldots, M_n$ are all non-empty and $C_1[\cdot]_{R_1}$ and $C_2[\cdot]_{R_2}$ are arbitrary, possibly empty contexts. Now, we can remove the occurrence of the connector P from the derivation $\mathcal{D}$ by replacing the subderivations above by

$$
\text{s.sw}\uparrow \frac{M_1 \otimes C_1\Big[C_2\big[P(\!|\varnothing, M_2, \ldots, M_n|\!)\big]_{R_2}\Big]_{R_1}}{C_2\Big[M_1 \invamp C_2\big[P(\!|\varnothing, M_2, \ldots, M_n|\!)\big]_{R_2}\Big]_{R_1}}
$$

or

$$
\text{s.sw}\downarrow \frac{C_1\Big[M_1 \otimes C_2\big[P(\!|\varnothing, M_2, \ldots, M_n|\!)\big]_{R_2}\Big]_{R_1}}{M_1 \invamp C_1\Big[C_2\big[P(\!|\varnothing, M_2, \ldots, M_n|\!)\big]_{R_2}\Big]_{R_1}}
$$

respectively.

(b) if $j \neq i$, then we can assume without loss of generality that $j = 1$ and $i = 2$, and we can (via the rule permutations in Figure 5) isolate a subderivation of one of the following two shapes

$$
\text{s.sw}\uparrow \frac{M_1 \otimes C_1\Big[C_2\big[P(\!|\varnothing, M_2, \ldots, M_n|\!)\big]_{R_2}\Big]_{R_1}}{C_1\left[\text{s.sw}\downarrow \dfrac{C_2\big[P(\!|M_1, \ldots, M_n|\!)\big]_{R_2}}{M_2 \invamp C_2\big[P(\!|M_1, \varnothing, M_3, \ldots, M_n|\!)\big]_{R_2}}\right]_{R_1}}
$$

or

$$
\text{s.sw}\downarrow \frac{C_1\left[\text{s.sw}\uparrow \dfrac{M_1 \otimes C_2\big[P(\!|\varnothing, M_2, \ldots, M_n|\!)\big]_{R_2}}{C_2\big[P(\!|M_1, \ldots, M_n|\!)\big]_{R_2}}\right]_{R_1}}{M_2 \invamp C_1\Big[C_2\big[P(\!|M_1, \varnothing, M_3, \ldots, M_n|\!)\big]_{R_2}\Big]_{R_1}}
$$

where the graphs $M_1, \ldots, M_n$ are non-empty and $C_1[\cdot]_{R_1}$ and $C_2[\cdot]_{R_2}$ are (possibly empty) contexts. We can replace these subderivations by

$$
\mathsf{s.sw}\!\uparrow \frac{M_1 \otimes C_1 \left[\mathsf{s.sw}\!\downarrow \dfrac{C_2 \big[P(\!|\varnothing, M_2, \ldots, M_n|\!) \big]_{R_2}}{M_2 \,\mathord{\mkern-1mu\invamp\mkern-1mu}\, C_2 \big[P(\!|\varnothing, \varnothing, M_3, \ldots, M_n|\!) \big]_{R_2}} \right]_{R_1}}{C_1 \left[M_2 \,\mathord{\mkern-1mu\invamp\mkern-1mu}\, C_2 \big[P(\!|M_1, \varnothing, M_3, \ldots, M_n|\!) \big]_{R_2} \right]_{R_1}}
$$

or

$$
\mathsf{s.sw}\!\downarrow \frac{C_1 \left[M_1 \otimes C_2 \big[P(\!|\varnothing, M_2, \ldots, M_n|\!) \big]_{R_2} \right]_{R_1}}{M_2 \,\mathord{\mkern-1mu\invamp\mkern-1mu}\, C_1 \left[\mathsf{s.sw}\!\downarrow \dfrac{M_1 \otimes C_2 \big[P(\!|\varnothing, \varnothing, M_3, \ldots, M_n|\!) \big]_{R_2}}{C_2 \big[P(\!|M_1, \varnothing M_3, \ldots, M_n|\!) \big]_{R_2}} \right]_{R_1}}
$$

respectively. In both cases, the connector P disappears, but can potentially be replaced by the a smaller connector, determined by $P(\!|\varnothing, \varnothing, M_3, \ldots, M_n|\!)$. However, in either case we can apply the induction hypothesis.

(2) An instance of $\mathsf{p}\!\downarrow$ destroys the connector P that has been introduced by a $\mathsf{s.sw}\!\downarrow$. This means that modulo rule permutations, we have a subderivation of the following shape (where we assume without loss of generality that $i = 1$):

$$
\mathsf{s.sw}\!\downarrow \frac{C \left[\mathsf{p}\!\downarrow \dfrac{(M_1 \,\mathord{\mkern-1mu\invamp\mkern-1mu}\, N_1) \otimes \cdots \otimes (M_n \,\mathord{\mkern-1mu\invamp\mkern-1mu}\, N_n)}{P(\!|M_1, \ldots, M_n|\!) \,\mathord{\mkern-1mu\invamp\mkern-1mu}\, P^{\perp}(\!|N_1, \ldots, N_n|\!)} \right]_R}{M_1 \,\mathord{\mkern-1mu\invamp\mkern-1mu}\, C \left[P(\!|\varnothing, M_2, \ldots, M_n|\!) \,\mathord{\mkern-1mu\invamp\mkern-1mu}\, P^{\perp}(\!|N_1, \ldots, N_n|\!) \right]_R}
$$

where $M_1 \neq \varnothing$. We can replace this subderivation by

$$
\mathsf{s.sw}\!\downarrow \frac{C \left[\mathsf{s.sw}\!\uparrow \dfrac{(M_1 \,\mathord{\mkern-1mu\invamp\mkern-1mu}\, N_1) \otimes \begin{array}{c} (M_2 \,\mathord{\mkern-1mu\invamp\mkern-1mu}\, N_2) \otimes \cdots \otimes (M_n \,\mathord{\mkern-1mu\invamp\mkern-1mu}\, N_n) \\ \mathcal{D}^* \,\big\|\, \text{Lemma 5.1} \\ P(\!|\varnothing, M_2, \ldots, M_n|\!) \,\mathord{\mkern-1mu\invamp\mkern-1mu}\, P^{\perp}(\!|\varnothing, N_2, \ldots, N_n|\!) \end{array}}{P(\!|\varnothing, M_2, \ldots, M_n|\!) \,\mathord{\mkern-1mu\invamp\mkern-1mu}\, P^{\perp}(\!|M_1 \,\mathord{\mkern-1mu\invamp\mkern-1mu}\, N_1, N_2, \ldots, N_n|\!)} \right]_R}{M_1 \,\mathord{\mkern-1mu\invamp\mkern-1mu}\, C \left[P(\!|\varnothing, M_2, \ldots, M_n|\!) \,\mathord{\mkern-1mu\invamp\mkern-1mu}\, P^{\perp}(\!|N_1, N_2, \ldots, N_n|\!) \right]_R}
$$

in which the connector P is no longer present. The derivation $\mathcal{D}^*$ exists by Lemma 5.1. Note that the construction of that lemma entails that for all connectors Q that occur in $\mathcal{D}^*$ and that do not already occur in M_j (for $j \in \{1, \ldots, n\}$) or the context, we have $|V_Q| < |V_P|$. We can therefore proceed by induction hypothesis. Note that we did not demand that $M_2, \ldots, M_n$ are non-empty, and therefore this case also applies when the roles of P and $P^{\perp}$ are exchanged.

(3) Finally, we have the case where an instance of $\mathsf{ai}\!\downarrow$ destroys the connector P. This happens when one of the modules M_j is $a \,\mathord{\mkern-1mu\invamp\mkern-1mu}\, a^{\perp}$ which is removed by the $\mathsf{ai}\!\downarrow$ instance. Again, we have two cases: $i = j$ and $i \neq j$.

(a) If $i = j$, then we can without loss of generality assume that $i = j = 1$. Modulo rule permutations, we have subderivation

$$\text{s.sw}\downarrow \frac{C\left[P\left(\left|\text{ai}\downarrow \frac{\varnothing}{a \,\Pbar\, a^\perp}\right., M_2, \ldots, M_n\right)\right]_R}{a \,\Pbar\, a^\perp \,\Pbar\, C\left[P(\varnothing, M_2, \ldots, M_n)\right]_R}$$

that can be replaced by

$$\text{ai}\downarrow \frac{\varnothing}{a \,\Pbar\, a^\perp} \,\Pbar\, C\left[P(\varnothing, M_2, \ldots, M_n)\right]_R$$

where the connector P does not occur anymore.

(b) If $i \neq j$, we can assume without loss of generality that $i = 2$ and $j = 1$, i.e., we have a subderivation

$$\text{s.sw}\downarrow \frac{C\left[P\left(\left|\text{ai}\downarrow \frac{\varnothing}{a \,\Pbar\, a^\perp}\right., M_2, \ldots, M_n\right)\right]_R}{M_2 \,\Pbar\, C\left[P(a \,\Pbar\, a^\perp, \varnothing, \ldots, M_n)\right]_R}$$

which can be replaced by

$$\text{s.sw}\downarrow \frac{C\left[P(\varnothing, M_2, \ldots, M_n)\right]_R}{M_2 \,\Pbar\, C\left[P\left(\left|\text{ai}\downarrow \frac{\varnothing}{a \,\Pbar\, a^\perp}\right., \varnothing, M_3, \ldots, M_n\right)\right]_R}$$

in which the connector P is no longer present (but which might contain a new connector Q that is a subconnector of P and has therefore smaller size). □

Corollary 8.4. *Let A be a cograph. If $\vdash_{\mathsf{GS} \cup \{\mathsf{s.sw}\uparrow\}} A$, then there is a derivation*

$$\begin{array}{c} \varnothing \\ \mathcal{D} \,\big\|\, \mathsf{GS} \cup \{\mathsf{s.sw}\uparrow\} \\ A \end{array} \tag{8.1}$$

such that every graph occurring in $\mathcal{D}$ is a cograph.

Proof. All subconnectors of a cograph are $\otimes$ or $\Pbar$. By Theorem 8.3 there is a derivation $\mathcal{D}$ of A in $\mathsf{GS} \cup \{\mathsf{s.sw}\uparrow\}$ in which only the connectors $\otimes$ and $\Pbar$ occur, i.e., every graph in $\mathcal{D}$ is a cograph. □

Using this result, we can now show that GS is a conservative extension of unit-free multiplicative linear logic with mix (denoted MLL°) [Gir87a]. The formulas of MLL° are as in (2.1), but without the unit, and the inference rules of MLL° are given in Figure 6, where Γ and Δ are *sequents*, i.e., multisets of formulas, separated by commas. We write $\vdash_{\mathsf{MLL}^\circ} \Gamma$ if the sequent Γ is provable in MLL°, i.e. if there is a finite proof tree in MLL° with conclusion Γ, where every leaf of the tree is an axiom.

$$\mathsf{ax}\frac{}{a, a^\perp} \qquad \otimes\frac{\Gamma, \phi \qquad \Delta, \psi}{\Gamma, \phi \otimes \psi, \Delta} \qquad \mathfrak{R}\frac{\Gamma, \phi, \psi}{\Gamma, \phi \,\mathfrak{R}\, \psi} \qquad \mathsf{mix}\frac{\Gamma \qquad \Delta}{\Gamma, \Delta}$$

Figure 6: The inference rules of the system MLL°

Lemma 8.5. *Let A and B be cographs. Then*

$$\mathsf{s.sw}{\downarrow}\frac{A}{B} \quad\Longrightarrow\quad \begin{array}{c}A\\\|\{\mathsf{sw}\}\\B\end{array} \qquad and \qquad \mathsf{s.sw}{\uparrow}\frac{A}{B} \quad\Longrightarrow\quad \begin{array}{c}A\\\|\{\mathsf{sw}\}\\B\end{array} \qquad (8.2)$$

Proof. By Theorem 2.6, the graphs A and B are cographs iff there are formulas ϕ and ψ with $[\![\phi]\!] \simeq A$ and $[\![\psi]\!] \simeq B$. Now the statement follows from the corresponding statement for formulas (see e.g., Lemma 4.3.20 in [Str03a]). □

Theorem 8.6. *Let A be a cograph. Then $\vdash_{\mathsf{GS}\cup\{\mathsf{s.sw}{\uparrow}\}} A$ iff $\vdash_{\{\mathsf{ai}{\downarrow},\mathsf{sw}\}} A$.*

Proof. The implication from right to left follows immediately from the fact that sw is a special case of both $\mathsf{s.sw}{\downarrow}$ and $\mathsf{s.sw}{\uparrow}$ (see Lemma 4.12). For the implication from left to right, apply Corollary 8.4 to get a derivation $\mathcal{D}$ that only uses cographs. Hence the rule $\mathsf{p}{\downarrow}$ is not used in $\mathcal{D}$. Therefore, by Lemma 8.5, we can get a derivation $\mathcal{D}'$ that only uses the rules $\mathsf{ai}{\downarrow}$ and sw. □

Corollary 8.7. *For any unit-free formula ϕ,*

$$\vdash_{\mathsf{MLL}^\circ} \phi \quad\Longleftrightarrow\quad \vdash_{\mathsf{GS}} [\![\phi]\!]$$

Proof. It has been shown before (see, e.g., [GS01, Str03a]) that a formula ϕ is provable in MLL° iff it is derivable using only the rules $\mathsf{ai}{\downarrow}$ and sw (when interpreted for formulas) modulo the equivalence $\equiv$, defined in (2.2). By Theorem 8.6, this is equivalent to having $[\![\phi]\!]$ provable in $\{\mathsf{ai}{\downarrow}, \mathsf{sw}\}$.[12] By Theorem 8.6 and Theorem 7.2 This is equivalent to $\vdash_{\mathsf{GS}} [\![\phi]\!]$. □

Corollary 8.8. *Provability in GS is **NP**-complete.*

Proof. Since MLL° is **NP**-complete, we can conclude from Corollary 8.7 that GS is **NP**-hard. Containment in **NP** has been shown in Observation 5.14. □

9. Graphs as Generalised Connectives

In the previous Section 8 we have shown that our system is a conservative extension of multiplicative linear logic with mix (MLL°): the logic MLL° is the restriction of GS to P_4-free graphs. Lemma 3.9 allows us to associate a modular decomposition-tree to a graph, in the same way as we associate a formula-tree to a cograph. This suggests the use of graphs as connectives in order to define *generalised formulas* that are encoding graphs. Our choices to name the two graphs on two vertices $\mathfrak{R}$ and $\otimes$ (see 3.2) and to denote the graph operations of union and join with the same symbols (Definition 2.3) are coherent with this intent. In fact, according with the notation for the composition-via-graph (Definition 3.8) we have

$$G \otimes H = \otimes(\![G, H]\!) \quad \text{and} \quad G \,\mathfrak{R}\, H = \mathfrak{R}(\![G, H]\!).$$

---

[12]Note that even though ϕ is unit-free, we allow units to occur in the derivation of ϕ, in particular, in (4.14) we can have $B = \varnothing$.

Definition 9.1 (Connective-as-graph). Any graph G with $|V_G| = n$ *describes* an n-ary connective G.

It should be clear that this notion of connective goes beyond the one of *synthetic connective*, which is a general connective definable as composition of standard binary connectives (see e.g. [And92, Gir00, MP13]). In this section we discuss the exact relation between our *connectives-as-graphs* notion with the notion of *(multiplicative) generalised connectives* from [Gir87b, DR89, Mai19, AM20]. In particular we show that our notion fits the definition of multiplicative connective, but it describes different mathematical objects from the ones known in the literature. For this, we first recall the notion of *multiplicative generalised connective* from the early work in linear logic [Gir87b, DR89], and their description as sets of partitions. Then we show that the two notions are different. More precisely, we show that

(1) our system proves different generalised formulas than MLL° extended with generalised connectives,
(2) the symmetries induced by our generalised connectives-as-graphs are different from the generalised connectives-as-partitions from [Gir87b, DR89, Mai19, AM20],
(3) the notion of *decomposable connective* differs in the two settings, and
(4) our generalised connectives-as-graphs do not suffer from the so called *packaging problem* [DR89].

Definition 9.2 (multiplicative generalised connective [Gir87b, DR89]). A *multiplicative generalised connective* is an n-ary connective C which admits a *linear* and *context-free* (introduction) rule in the sequent calculus, that is, a rule of the form

$$\frac{\vdash \Gamma_1, \phi_1, \ldots, \phi_{k_1} \quad \cdots \quad \vdash \Gamma_k, \phi_{1+i_{k-1}}, \ldots, \phi_n}{\vdash \Gamma, \mathsf{C}(\phi_{\sigma(1)}, \ldots, \phi_{\sigma(n)})} \quad \begin{array}{l} \text{with } \Gamma = \Gamma_1 \uplus \cdots \uplus \Gamma_k \\ \text{and } \sigma \text{ a permutation of } \{1, \ldots, n\} \end{array} \quad (9.1)$$

The linearity condition demands that each occurrence of a formula in Γ or sub-formula of the *active formula* $\mathsf{C}(\phi_{\sigma(1)}, \ldots, \phi_{\sigma(n)})$ occurs exactly once in the premise, while the context-freeness condition demands that the application of the rule does not depend on the shape of the premises.

By means of example, consider the connectives of linear logic. The multiplicative conjunction $\otimes$ and $\invamp$ disjunction are multiplicative connectives, while the additive conjunction & and disjunction $\oplus$ of linear logic are not.

$$\otimes \frac{\vdash \Gamma_1, \phi \quad \vdash \Gamma_2, \psi}{\vdash \Gamma, \phi \otimes \psi} \text{ where } \Gamma = \Gamma_1 \uplus \Gamma_2 \quad \invamp \frac{\vdash \Gamma, \phi, \psi}{\vdash \Gamma, \phi \invamp \psi} \quad \& \frac{\vdash \Gamma, \phi \quad \vdash \Gamma, \psi}{\vdash \Gamma, \phi \mathbin{\&} \psi} \quad \oplus \frac{\vdash \Gamma, \phi}{\vdash \Gamma, \phi \oplus \psi}$$

In fact, each formula in Γ and the two sub-formulas ϕ and ψ of the active formula occur exactly once in the premises of the rules for $\otimes$ and $\invamp$. On the contrary, the rule for & is not context-free since it can be applied only if the premises are of the form Γ, ϕ and Δ, ψ with $\Gamma = \Delta$, that is, the rule depends on the context. Moreover this rule is not linear since each occurrence of a formula in Γ occurs twice in the premises. The rule for $\oplus$ is not linear since it produces a sub-formula ψ which does not occur in the premise.

Following Definition 9.2, our connectives-as-graphs (Definition 9.1) are multiplicative connectives. In fact, the n-ary connective described by a graph G with $|V_G| = n$ admits a unique sequent calculus (introduction) rule of the form

$$G \frac{\vdash \Gamma_1, \phi_1 \quad \cdots \quad \vdash \Gamma_n, \phi_n}{\vdash \Gamma, G(\!(\phi_1, \ldots, \phi_n)\!)} \quad \text{with } \Gamma = \Gamma_1 \uplus \cdots \uplus \Gamma_n$$

as it is derivable in GS if we interpret formulas as graphs and sequents of formulas as their disjoint union, and "premises concatenation" as the joint of the sequents in the premises, that is

$$G\,\frac{([\![\Gamma_1]\!]\,\mathbin{⅋}\,[\![\phi_1]\!])\otimes\cdots\otimes([\![\Gamma_n]\!]\,\mathbin{⅋}\,[\![\phi_n]\!])}{[\![\Gamma_1]\!]\,\mathbin{⅋}\cdots\mathbin{⅋}\,[\![\Gamma_n]\!]\,\mathbin{⅋}\,G([\![\phi_1]\!],\ldots,[\![\phi_n]\!])} \quad \sim \quad \begin{array}{c} ([\![\Gamma_1]\!]\,\mathbin{⅋}\,[\![\phi_1]\!])\otimes\cdots\otimes([\![\Gamma_n]\!]\,\mathbin{⅋}\,[\![\phi_n]\!]) \\ \mathcal{D}\,\|\,\mathsf{GS} \\ \text{s.sw}{\downarrow}\,\dfrac{G^{\perp}([\![\Gamma_1]\!],\ldots,[\![\Gamma_n]\!])\,\mathbin{⅋}\,G([\![\phi_1]\!],\ldots,[\![\phi_n]\!])}{[\![\Gamma_1]\!]\,\mathbin{⅋}\cdots\mathbin{⅋}\,[\![\Gamma_n]\!],\mathbin{⅋}G([\![\phi_1]\!],\ldots,[\![\phi_n]\!])} \end{array}$$

where $\mathcal{D}$ exists by Lemma 5.1.

Let us now recall the description of generalised connectives by means of *sets of partitions* from [DR89, Mai19, AM20]. We denote by $\mathbb{P}_n$ the set of partitions of the set $\{1,\ldots,n\}$ and we say that an n-ary connective is described by a non-empty subset of $\mathbb{P}_n$. To improve readability, we follow the following policy for parenthesis: $\{\,\}$ for sets (of partitions), $\langle\,\rangle$ for partitions (i.e. family of disjointed covering sets of subsets of $\{1,\ldots,n\}$) and $[\,]$ for the elements of each partition (i.e. subsets of $\{1,\ldots,n\}$). Each of these partitions can be interpreted intuitively as a way in which it is possible to gather the sub-formulas of the active formula into the premises of a multiplicative rule. By means of example, the binary rules for $\otimes$ and $\mathbin{⅋}$ are described respectively by the (singleton) partition sets $\{\langle[1],[2]\rangle\}$ and $\{\langle[1,2]\rangle\}$. If we define a generalised (synthetic) connective $\mathsf{C}(a,b,c)=(a\otimes b)\mathbin{⅋} c$, then we can associate to it the multiple rules corresponding to all possible ways to produce this formula in multiplicative linear logic:

$$\mathbin{⅋}\frac{\otimes\dfrac{\vdash a,c\quad\vdash b}{\vdash(a\otimes b),c}}{\vdash(a\otimes b)\mathbin{⅋} c}\leadsto\mathsf{C}^{[1,3],[2]}\dfrac{1,3\quad 2}{\mathsf{C}(1,2,3)}\leadsto\langle[1,3],[2]\rangle \quad\Bigg|\quad \mathbin{⅋}\frac{\otimes\dfrac{\vdash a\quad\vdash b,c}{\vdash(a\otimes b),c}}{\vdash(a\otimes b)\mathbin{⅋} c}\leadsto\mathsf{C}^{[1],[2,3]}\dfrac{1\quad 2,3}{\mathsf{C}(1,2,3)}\leadsto\langle[1],[2,3]\rangle$$

Because of De Morgan duality, we need for every connective a dual connective. The interactions of dual connectives by means of the cut-rule, together with the resulting cut-elimination procedure, enforces some additional structure between the sets of partitions. Given two partitions $p,q\in\mathbb{P}_n$, we define their *incidence graph* as the multi-graph whose vertices are the elements of p and q, such that there is an edge between two different vertices for every element in their intersection. For example, we have the following incidence graphs between $p=\langle[1,3],[2]\rangle$ and each of $q_1=\langle[1],[2],[3]\rangle$, $q_2=\langle[1],[2,3]\rangle$ and $q_3=\langle[1,2,3]\rangle$:

We say that two partitions $p,q\in\mathbb{P}_n$ are *orthogonal* if their incidence graph is connected and acyclic. In the example above, p and q_2 are orthogonal, but p and q_1 are not (because the incidence graph is not connected) and p and q_3 are not (because the incidence graph is not acyclic). We say that $P,Q\subseteq\mathbb{P}_n$ are *orthogonal* if every partition in P is orthogonal to every partition in Q.

Definition 9.3 (Connective-as-partitions)**.** Let P be a set of partitions in $\mathbb{P}_n$ such that there is a set of partitions $Q\subseteq\mathbb{P}_n$ that is orthogonal to P. An n-ary connective C *is described* by P if C admits exactly one multiplicative inference rule C^p of shape (9.1) for each $p\in P$,

such that any two subformulas ϕ_i and ϕ_j of the principal formula $\mathsf{C}(\phi_1, \ldots, \phi_n)$ belong to the same premise iff i and j belong to a same element in p.

For the purpose of this section, we consider the pair of dual generalised 4-ary connectives first introduced in [Gir87b] by means of permutations and later reformulated in [DR89] by means of partitions. Following the formalism in [AM20], we denote by $\mathsf{G_4}$ and $\mathsf{G_4^\perp}$ the connectives respectively described by the following two sets of partitions in $\mathbb{P}_4$

$$\mathsf{G_4}: \{\langle [1,2], [3,4] \rangle, \langle [1,4], [2,3] \rangle\} \quad \text{and} \quad \mathsf{G_4^\perp}: \{\langle [1,3], [2], [4] \rangle, \langle [2,3], [1], [3] \rangle\} \tag{9.2}$$

That is, for the connective $\mathsf{G_4}$ we have the following two inference rules

$$\mathsf{G_4}^{[1,2],[3,4]} \frac{\Gamma, \phi, \psi \qquad \Delta, \rho, \chi}{\Gamma, \Delta, \mathsf{G_4}(\phi, \psi, \rho, \chi)} \qquad \mathsf{G_4}^{[1,4],[2,3]} \frac{\Gamma, \phi, \chi \qquad \Delta, \psi, \rho}{\Gamma, \Delta, \mathsf{G_4}(\phi, \psi, \rho, \chi)} \tag{9.3}$$

Let $\mathcal{C}$ be the set of non-decomposable (multiplicative) connectives-as-partitions. We call a *generalised formula* a formula that is generated by a countable set of positive or negative propositional atoms $\{a, a^\perp, b, b^\perp, \ldots\}$ via the following grammar:

$$\phi, \psi ::= a \mid a^\perp \mid \phi \,\mathbin{⅋}\, \psi \mid \phi \otimes \psi \mid \mathsf{C}(\phi_1, \ldots, \phi_n)$$

where $\mathsf{C} \in \mathcal{C}$ is any n-ary connective. We denote by $\mathsf{MLL}_\mathcal{C}^\circ$ the proof system over generalised formulas extending MLL° with, for each $\mathsf{C} \in \mathcal{C}$, all the sequent rules of C.

Comparing Theorems. We can now prove that $\mathsf{MLL}_\mathcal{C}^\circ$ and GS deal with different objects by showing that the connective $\mathsf{G_4}$ does not correspond to any instance of the connective $\mathsf{P_4}$, which is described by the only prime graph with 4 vertices. From now on, we refer to $\mathsf{P_4}(a, b, c, d)$ as the graph $a{—}b{—}c{—}d$.

Proposition 9.4. *In $\mathsf{MLL}_\mathcal{C}^\circ$, none of the following formulas or their equivalent sequents are provable:*

$c \otimes (d \,\mathbin{⅋}\, (a \otimes b)) \multimap \mathsf{G_4}(a, b, c, d)$	$\mathsf{G_4}(a, b, c, d), c^\perp \,\mathbin{⅋}\, (d^\perp \otimes (a^\perp \,\mathbin{⅋}\, b^\perp))$
$d \otimes (c \,\mathbin{⅋}\, (a \otimes b)) \multimap \mathsf{G_4}(a, b, c, d)$	$\mathsf{G_4}(a, b, c, d), d^\perp \,\mathbin{⅋}\, (c^\perp \otimes (a^\perp \,\mathbin{⅋}\, b^\perp))$
$a \otimes (c \,\mathbin{⅋}\, (b \otimes d)) \multimap \mathsf{G_4}(a, b, c, d)$	$\mathsf{G_4}(a, b, c, d), a^\perp \,\mathbin{⅋}\, (c^\perp \otimes (b^\perp \,\mathbin{⅋}\, d^\perp))$
$c \otimes (a \,\mathbin{⅋}\, (b \otimes d)) \multimap \mathsf{G_4}(a, b, c, d)$	$\mathsf{G_4}(a, b, c, d), c^\perp \,\mathbin{⅋}\, (a^\perp \otimes (b^\perp \,\mathbin{⅋}\, d^\perp))$
$a \otimes (d \,\mathbin{⅋}\, (b \otimes c)) \multimap \mathsf{G_4}(a, b, c, d)$	$\mathsf{G_4}(a, b, c, d), a^\perp \,\mathbin{⅋}\, (d^\perp \otimes (b^\perp \,\mathbin{⅋}\, c^\perp))$
$d \otimes (a \,\mathbin{⅋}\, (b \otimes c)) \multimap \mathsf{G_4}(a, b, c, d)$	$\mathsf{G_4}(a, b, c, d), d^\perp \,\mathbin{⅋}\, (a^\perp \otimes (b^\perp \,\mathbin{⅋}\, c^\perp))$

$$\tag{9.4}$$

Proof. We show that the first sequent is not provable. The same reasoning applies to the other sequents. In a derivation of $\mathsf{G_4}(a, b, c, d), c^\perp \,\mathbin{⅋}\, (d^\perp \otimes (a^\perp \,\mathbin{⅋}\, b^\perp))$ we can apply as first rule a $\mathbin{⅋}$. Any derivation continuing with a $\otimes$ rule must have in one branch $\mathsf{G_4}(a, b, c, d)$ and either $d^\perp$ or $a^\perp \,\mathbin{⅋}\, b^\perp$. This makes it impossible to match all pairs of dual atoms in axioms. Similarly, starting with one of the two $\mathsf{G_4}$ rules in 9.3, or starting with the $\mathbin{⅋}$ rule followed by one of these two rules, we encounter the same problem of mismatched atoms. $\square$

Proposition 9.5. *In* GS*, the graphs corresponding to the following formulas are provable:*

$$
\begin{array}{c|c}
c \otimes (d \,\mathfrak{N}\, (a \otimes b)) \multimap \mathsf{P}_4(\!|a,b,c,d|\!) & \mathsf{P}_4(\!|a,b,c,d|\!) \,\mathfrak{N}\, c^\perp \,\mathfrak{N}\, (d^\perp \otimes (a^\perp \,\mathfrak{N}\, b^\perp)) \\
c \otimes (d \,\mathfrak{N}\, (a \otimes b)) \multimap \mathsf{P}_4(\!|b,a,c,d|\!) & \mathsf{P}_4(\!|b,a,c,d|\!) \,\mathfrak{N}\, c^\perp \,\mathfrak{N}\, (d^\perp \otimes (a^\perp \,\mathfrak{N}\, b^\perp)) \\ \hline
d \otimes (c \,\mathfrak{N}\, (a \otimes b)) \multimap \mathsf{P}_4(\!|b,a,d,c|\!) & \mathsf{P}_4(\!|b,a,d,c|\!) \,\mathfrak{N}\, d^\perp \,\mathfrak{N}\, (c^\perp \otimes (a^\perp \,\mathfrak{N}\, b^\perp)) \\
d \otimes (c \,\mathfrak{N}\, (a \otimes b)) \multimap \mathsf{P}_4(\!|a,b,d,c|\!) & \mathsf{P}_4(\!|a,b,d,c|\!) \,\mathfrak{N}\, d^\perp \,\mathfrak{N}\, (c^\perp \otimes (a^\perp \,\mathfrak{N}\, b^\perp)) \\ \hline
a \otimes (c \,\mathfrak{N}\, (b \otimes d)) \multimap \mathsf{P}_4(\!|c,a,d,b|\!) & \mathsf{P}_4(\!|c,a,d,b|\!) \,\mathfrak{N}\, a^\perp \,\mathfrak{N}\, (c^\perp \otimes (b^\perp \,\mathfrak{N}\, d^\perp)) \\
a \otimes (c \,\mathfrak{N}\, (b \otimes d)) \multimap \mathsf{P}_4(\!|c,a,b,d|\!) & \mathsf{P}_4(\!|c,a,b,d|\!) \,\mathfrak{N}\, a^\perp \,\mathfrak{N}\, (c^\perp \otimes (b^\perp \,\mathfrak{N}\, d^\perp)) \\ \hline
c \otimes (a \,\mathfrak{N}\, (b \otimes d)) \multimap \mathsf{P}_4(\!|a,c,b,d|\!) & \mathsf{P}_4(\!|a,c,b,d|\!) \,\mathfrak{N}\, c^\perp \,\mathfrak{N}\, (a^\perp \otimes (b^\perp \,\mathfrak{N}\, d^\perp)) \\
c \otimes (a \,\mathfrak{N}\, (b \otimes d)) \multimap \mathsf{P}_4(\!|a,c,d,b|\!) & \mathsf{P}_4(\!|a,c,d,b|\!) \,\mathfrak{N}\, c^\perp \,\mathfrak{N}\, (a^\perp \otimes (b^\perp \,\mathfrak{N}\, d^\perp)) \\ \hline
a \otimes (d \,\mathfrak{N}\, (b \otimes c)) \multimap \mathsf{P}_4(\!|c,b,a,d|\!) & \mathsf{P}_4(\!|c,b,a,d|\!) \,\mathfrak{N}\, a^\perp \,\mathfrak{N}\, (d^\perp \otimes (b^\perp \,\mathfrak{N}\, c^\perp)) \\
a \otimes (d \,\mathfrak{N}\, (b \otimes c)) \multimap \mathsf{P}_4(\!|b,c,a,d|\!) & \mathsf{P}_4(\!|b,c,a,d|\!) \,\mathfrak{N}\, a^\perp \,\mathfrak{N}\, (d^\perp \otimes (b^\perp \,\mathfrak{N}\, c^\perp)) \\ \hline
d \otimes (a \,\mathfrak{N}\, (b \otimes c)) \multimap \mathsf{P}_4(\!|a,d,c,b|\!) & \mathsf{P}_4(\!|a,d,c,b|\!) \,\mathfrak{N}\, d^\perp \,\mathfrak{N}\, (a^\perp \otimes (b^\perp \,\mathfrak{N}\, c^\perp)) \\
d \otimes (a \,\mathfrak{N}\, (b \otimes c)) \multimap \mathsf{P}_4(\!|a,d,b,c|\!) & \mathsf{P}_4(\!|a,d,b,c|\!) \,\mathfrak{N}\, d^\perp \,\mathfrak{N}\, (a^\perp \otimes (b^\perp \,\mathfrak{N}\, c^\perp))
\end{array}
\tag{9.5}
$$

Proof. The proof of $(d \,\mathfrak{N}\, (b \otimes (a \,\mathfrak{N}\, c))) \multimap \mathsf{P}_4(\!|c,a,d,b|\!)$ is shown in (4.13). The other implications are proven similarly. □

Both $\mathsf{MLL}_{\mathcal{C}}^{\circ}$ (by definition) and GS (by Corollary 8.7) are conservative extensions of MLL. The non $\otimes/\mathfrak{N}$-decomposable 4-ary connective G_4 should be translated as a non-decomposable graph with 4 vertices, that is, a P_4. Thus, the translation of $\mathsf{G}_4(a,b,c,d)$ should be of the shape $\mathsf{P}_4(\!|\sigma(a),\sigma(b),\sigma(c),\sigma(d)|\!)$ for some permutation σ over the set $\{a,b,c,d\}$. However for each possible translation of G_4, one of the implications in (9.4), which are not derivable in $\mathsf{MLL}_{\mathcal{C}}^{\circ}$ and contain G_4, is translated into a GS-derivable implication. We can conclude that the systems $\mathsf{MLL}_{\mathcal{C}}^{\circ}$ and GS are not equivalent.

Symmetries of a connective. We now calculate the symmetries of the connectives G_4 and P_4 and show that they are different in a way such that GS and $\mathsf{MLL}_{\mathcal{C}}^{\circ}$ are not comparable as logical systems. To understand what we mean by symmetries of a connective, consider for example the $\otimes$, which is commutative—as the formula $A \otimes B$ is logically equivalent to $B \otimes A$—that is, we can permute A and B without changing the formula. On the other hand, the connective $\multimap$ is not commutative. But we can define a connective $\circ\!\!-$ distinct from $\multimap$ but definable from $\multimap$ as $A \circ\!\!- B = B \multimap A$.

Once we start to analyse n-ary connectives with $n > 2$, we can no more categorise connectives only as commutative and non-commutative. Consider the 4-ary connectives G_4 and $\mathsf{G}_4^\perp$ and their describing sets of partitions in 9.2. We have

$$
\begin{aligned}
\mathsf{G}_4(a,b,c,d) = \mathsf{G}_4(b,a,c,d) = \mathsf{G}_4(a,b,d,c) = \mathsf{G}_4(b,c,d,a) = \\
\mathsf{G}_4(c,d,a,b) = \mathsf{G}_4(d,a,b,c) = \mathsf{G}_4(b,a,d,c) = \mathsf{G}_4(d,c,b,a)
\end{aligned}
$$

which, using interaction net syntax [Laf95], could be represented as follows

Hence, the sets of partitions describing $\mathsf{G_4}$ and $\mathsf{G_4^\perp}$ are *stable* under the permutation[13] in the set $\mathfrak{S}(\mathsf{G_4})$, shown below:

$$\mathfrak{S}(\mathsf{G_4}) = \mathfrak{S}(\mathsf{G_4^\perp}) = \{(1), (1,2), (3,4), (1,2,3,4), (1,3)(2,4), (1,4,2,3), (1,2)(3,4), (1,4)(2,3)\}$$

We conclude that there are $\frac{|\mathfrak{S_4}|}{|\mathfrak{S}(\mathsf{G_4})|} = 3$ non-isomorphic instances of $\mathsf{G_4}$, defining as many 4-ary connectives:

Moreover, we observe that $\mathsf{G_4^\perp}$ cannot be expressed as a function of $\mathsf{G_4}$ since the two describing sets are made of partitions of different cardinality. The connective $\mathsf{G_4}$ is a non-decomposable 4-ary connective [Mai19, AM20]. We can conclude that there are only three pairs of dual non-decomposable 4-ary connectives-as-partitions. In fact, a properly defined duality forces the dual of a set P of partitions to be exactly the set all partitions that are orthogonal to the ones in P. This restrains the subsets of $\mathbb{P}_4$ which can be used to describe connectives, leaving only 6 non-decomposable connectives.

At the same time, $\mathsf{P_4}$ is the unique prime graph on four vertices. Hence, any non-decomposable 4-ary connective-as-graph has to be an instance of $\mathsf{P_4}$. Since the symmetry group of $\mathsf{P_4}$ (i.e., the group of its isomorphisms) is the following

$$\mathfrak{S}(\mathsf{P_4}) = \{(1), (1,4)(2,3)\}$$

we conclude that there are $\frac{|\mathfrak{S_4}|}{|\mathfrak{S}(\mathsf{P_4})|} = 12$ different instances of $\mathsf{P_4}$ defining as many 4-ary connectives:

If we denote $\mathsf{Z}(\!(a,b,c,d)\!) = \mathsf{P_4}(\!(a,b,c,d)\!)$ and $\mathsf{N}(\!(a,b,c,d)\!) = \mathsf{P_4}(\!(c,a,d,b)\!)$ (the first and second-to-last graph in the first line in the above figure), we have that $\mathsf{Z}^\perp = \mathsf{N}$. More generally, if $\mathsf{C}(\!(a,b,c,d)\!) = \mathsf{P_4}(\!(\sigma(a),\sigma(b),\sigma(c),\sigma(d))\!)$ for a permutation σ of the set $\{a,b,c,d\}$, then

We conclude that there are 6 pairs of dual non-decomposable 4-ary connectives-as-graphs.

Summing up, the symmetry group of $\mathsf{G_4}$ is different from the one of $\mathsf{P_4}$. Hence, the two connectives have to be distinct. Moreover $\mathsf{G_4^\perp}(a,b,c,d)$ cannot be expressed as $\mathsf{G_4}(\sigma(a),\sigma(b),\sigma(c),\sigma(d))$ for any σ permutation over $\{a,b,c,d\}$, while $\mathsf{P_4}(\!(a,b,c,d)\!) = $

---

[13]We write permutations using the *cycle notation*, that is, the permutation is written as a product of cycles, and each element is mapped to the following element in the same cycle.

$P_4{}^\perp(\!|b, d, a, c|\!)$. In $\mathsf{MLL}_{\mathcal{C}}^{\circ}$ there are 3 pairs of dual non-decomposable 4-ary connectives, in GS there are 6 of such pairs.

Decomposable and non-decomposable connectives. In the connectives-as-partitions setting, a generalised n-ary connective described by a partition $P \subset \mathbb{P}_n$ is *decomposable* if there is a $\otimes/\mathbin{\invamp}$-formula F such that P is the set of partitions associated to all possible derivations of F [DR89, AM20]. For example, the connective described by $P = \{\langle[1,2],[2]\rangle, \langle[1],[2,3]\rangle\}$ is decomposable since it is associated to the formula $F = (a \otimes b) \mathbin{\invamp} c$ as shown above. However, the generalised connective which corresponds to the formula $\mathsf{G_4}(1, 2, 3, \mathsf{G_4}(4, 5, 6, 7))$ is considered to be non-decomposable, even if we define it by using other connectives. A possible motivation for such a choice may be the lack of an efficient decomposition algorithm in the literature. In fact, even from the set of partitions defined by a $\otimes/\mathbin{\invamp}$-formula, it is not trivial to reconstruct the original formula.

However, for our connectives-as-graphs setting, Lemma 3.9 provides a finer definition of connective decomposition. In particular, every prime graph defines a non-decomposable connective, and every connective is either non-decomposable or admits a decomposition into non-decomposable ones. That is, while the notion of decomposition for connectives-as-partitions is still rough, the one for connectives-as-graph is well studied and comes with a linear decomposition algorithm [HdMP04].

The packaging problem. The so called *packaging problem* [DR89] is the impossibility of deriving the axiom $\mathsf{G_4} \multimap \mathsf{G_4}$ in the sequent calculus. This is due to the lack of the *initial coherence* property [AL01, MP13] for the $\mathsf{MLL}_{\mathcal{C}}^{\circ}$ sequent calculus. This can easily be shown in the case of $\mathsf{G_4}$ by remarking that the arities of the rules for $\mathsf{G_4}$ and $\mathsf{G_4^{\perp}}$ make it impossible to gather together the premises as four smaller proofs. The initial coherence property can be recovered using the proof net syntax of $\mathsf{MLL}_{\mathcal{C}}^{\circ}$ [AM20].

As we have seen in Corollary 5.2, the proof system GS has the initial coherence property, that is, there is no packaging problem in GS.

10. Related and Future Work

In this section we would like to draw attention to certain challenges and open problems surrounding GS. Using examples, such as (6.6) and (6.4), we have already explained why GS necessarily demands deep inference. Let us now explain that simply taking an established semantics for MLL° based on graphs and dropping the restriction to cographs does not immediately yield a semantics for GS.

10.1. **Linear inferences in classical logic.** The switch rule has the property that it reflects edges and maximal cliques. That is: if there is an edge in the conclusion it will also appear in the premise and every maximal clique in the premise is a superset of some maximal clique in the conclusion. Indeed, mappings reflecting maximal cliques and preserving stable sets (mutually independent vertices) have a long history in program semantics [Ber78] which led to coherence spaces and the discovery of linear logic [Gir87a]. Therefore it is a reasonable starting point to try generalising switch by using such maximal clique reflecting homomorphisms, instead of $\mathsf{s.sw}\!\downarrow$. Indeed this is how we discovered $\mathsf{s.sw}\!\downarrow$, which is sound with respect to such homomorphisms.

Unfortunately, replacing s.sw↓ with maximal clique reflecting homomorphisms yields a system distinct from our graphical system, for example the following would be provable, but is not provable in GS.

$$
\begin{array}{cc}
a\!\!-\!\!b \\
a^{\perp}\;\;\;\;\;\;b^{\perp} \\
c\;\;\;\;c^{\perp}
\end{array}
\tag{10.1}
$$

We may try replacing both s.sw↓ and s.sw↑ using a stronger symmetric notion of homomorphism where, in addition, every maximal stable set in the conclusion is a superset of some maximal stable set in the premise. Homomorphisms that are both maximal clique reflecting and stable set preserving are special *linear inferences* [DS15, DS16] that preserve edges from the conclusion in the premise of a rule, which we refer to as *linear homomorphisms* for convenience. Using linear homomorphisms, the above example is not provable. To see why, observe that at some point either a and $a^{\perp}$ or b and $b^{\perp}$ must be brought together into a module where they can interact, but this cannot be achieved while preserving the maximal stable set $\{a, b^{\perp}\}$.

Notice, however, that if we made the alternative design decision of replacing s.sw↓ and s.sw↑ by linear homomorphisms described above, the implications below would be provable.

$$
\begin{array}{cc}
a\;\;\;\;e \\
b\!\!-\!\!d \\
c
\end{array}
\;\multimap\;
\begin{array}{cc}
a\;\;\;\;e \\
b\!\!-\!\!d \\
c
\end{array}
\qquad\qquad
\begin{array}{c}
a \\
b \\
c \\
d \\
e
\end{array}
\;\multimap\;
\begin{array}{c}
a \\
b \\
c \\
d \\
e
\end{array}
\tag{10.2}
$$

In contrast, the above examples are not provable in GS, since both sides are distinct prime graphs; and there is no suitable way to apply s.sw↓. Thus, we would obtain a distinct system from GS by using such linear homomorphisms.

The above examples separating linear homomorphisms from GS made use of graphs with at least 5 nodes that are not cographs. The smallest examples involving cographs only that separate MLL° from logics based on linear homomorphisms involve formulas with 8 nodes.[14] Indeed, studying logics defined using linear inferences that reflect maximal cliques and preserve maximal stable sets is currently a topic of active research and leads to possible extensions of Boolean logic to graphs [Cal16, War19, CDW20, Das19, DR21].

Logics resulting from directly generalising linear inferences inspired by Boolean logic are incomparable to GS. In one direction, the above examples show there are graphs that are not provable in GS that are provable using linear inferences; while in the converse direction the p↓ rule is not admissible in a system with linear inferences. To see why, observe that in Example 4.16 there is a clique $\{a^{\perp}, d^{\perp}, c, b\}$ in the premise of the p↓ that is not reflected in the conclusion. Since p↓ is necessary for examples such as in Figure 2, we know that the logics are incomparable. Hence, although these proposed logics on graphs appear to be close to GS, they are founded on logical principles that are different from the ones we assume, as explained in detail in Appendix B.

---

[14]A new linear inference of size 8: `https://prooftheory.blog/2020/06/25/new-linear-inference/` and Linear inferences of size 7: `https://prooftheory.blog/2020/10/01/linear-inferences-of-size-7/`

10.2. **Criteria for proof nets.** Graphical approaches to proof nets such as $R\&B$-graphs [Ret03] have valid definitions when we drop the restriction to cographs. However, we show that (at least without strengthening established criteria), such definitions do not yield a semantics for a logic over graphs, since logical principles laid out in the introduction are violated.

Consider again graph (4.3), which is not provable in GS. In an $R\&B$-graph we draw blue edges representing the axiom links of proof nets, as shown below for graph (4.3).

$$a^{\perp}\!\!-\!\!-a$$
$$\begin{matrix} | & \diagdown & | \\ b\!\!-\!\!-b^{\perp} \end{matrix}$$

(10.3)

If we remove the restriction to cographs when we apply the established correctness criterion for $R\&B$-graphs the above graph would be accepted, which is something we aim to avoid for a semantics of GS. The reason is the cycle of 4 vertices alternating between red and blue edges has a chord. This observation is independent of the rules of the system GS, as we observed in Section 4 that graph (4.3) cannot be provable in a system subject to the logical principle of consistency. Future work includes defining a stronger criterion for $R\&B$-graphs that works for graphs that are not cographs, with the aim of obtaining a sound and complete semantics for GS.

10.3. **Beyond formulas-as-processes.** As mentioned in the introduction, this study originates from the limited expressive power of formulas in the formulas-as-processes interpretation. In order to study the challenge of moving from formulas to graphs we have restricted ourselves to a minimal logic, whereas applications in concurrency typically require a logic with more features such as non-commutative operators for modelling the causal order of events [Bru02, Gug07, Ret99]. In the setting of graphical logics extending GS, non-commutative operators generalise to graphs incorporating directed edges which capture more general patterns of logical time, as explained in [AHMS22]. To make such models useful we also require features such as: additives for choices [TM05, Hor15]; quantifiers for dealing with message passing and private data [HTAC19, HT19]; and even co-inductive proofs [Hor20]. Lifting such features to the setting of graphs seem feasible, but requires a certain amount of future work.

To hint at further possibilities enabled by GS in the direction of concurrency theory, we return briefly to graph (1.2) mentioned in the introduction. We explained there that an edge models separation, the absence of an edge models the ability to communicate, and that we can express intransitive information flows using such graphs. We elaborate a little more on such a scenario below.

Suppose that the processes a, b, c, and d operate on three shared resources x, y, and z such that c and a operate (read and write) on x; a and d operate on y; and d and b operate on z. In addition, in this concurrency scenario, we assume that the information flow is strictly managed such that, although information can flow via x from c to a, and from a to d via y, those information flows must be strictly separated, and therefore a must be trusted or designed such that the information from c does not flow to d indirectly via a. More concretely, an illegal indirect information flow caused by a failing to manage correctly the separation between c and d may be achieved by the following series of read and write operations: assuming secret is known initially by c only, c writes secret to x, a reads secret from x and then writes secret to y, and then d reads secret from y. Similarly, d must be

careful to ensure no information is leaked between b and a, despite having a shared channel with both. Notice that since we should also avoid information flows between b and c, both d and a must trust each other to adhere to policy in order to manage their own policy obligations.

This is a realistic information flow scenario. For example a and d, may represent actions of service providers (e.g., a reviewer and editor) that make decisions based on confidential information from multiple parties (authors, reviewers, editors, and publishers) without disclosing confidential information to the other parties in a chain of conflicts of interest, and also without mutually sharing all information. From this small example, we see already that intransitive information flows arise naturally in networks where privacy is important; that they model complex decentralised trust scenarios; and hence, we anticipate further work in this direction.

11. Conclusion

Guided by logical principles, we have devised a minimal proof system (called GS and shown in Figure 1) that operates directly over graphs, rather than formulas. Negation is given by graph duality, while disjunction is disjoint union of graphs, allowing us to define the implication "G implies H" as the standard "not G or H" (see Definition 4.1). All other design decisions are then fixed by our guiding logical principles (Appendix B). Most of these principles follow from cut elimination (Theorem 5.4), to which the majority of this paper is dedicated. We also confirm that GS conservatively extends MLL° (Corollary 8.7) — a logic at the core of many proof systems.

Surprisingly, even for such a minimal generalisation of logic to graphs, deep inference is necessary. Proof systems for classical logic, intuitionistic logic, linear logic, and many other logics *may* be expressed using deep inference, but deep inference is generally not necessary, as presentations in the sequent calculus do exist. In contrast, for some logics (e.g., BV [Gug07, Tiu06] and modal logic S5 [Sto07, Pog08]), deep inference is necessary in order to define a proof system satisfying cut elimination. System GS goes further than the aforementioned systems in that all intermediate lemmas such as splitting (Lemmas 6.3 and 6.5) and context reduction (Lemma 6.6) demand a deep formulation that is more context aware than standard. As such we were required to generalise the basic mechanisms of deep inference itself in order to establish cut elimination (Theorem 5.4) for a logic over graphs. This need for deep inference for GS is due to a property of general graphs that is forbidden for graphs corresponding to formulas — that the shortest path between any two connected vertices may be greater than two; and hence, when we apply reasoning inside a module (i.e., a context), there may exist dependencies that indirectly constrain the module.

References

[AG21]   Matteo Acclavio and Giulio Guerrieri. A deep inference system for differential linear logic. *Electronic Proceedings in Theoretical Computer Science*, 353:26–49, dec 2021. URL: `https://doi.org/10.4204%2Feptcs.353.2`, `doi:10.4204/eptcs.353.2`.

[AHMS22] Matteo Acclavio, Ross Horne, Sjouke Mauw, and Lutz Straßburger. A graphical proof theory of logical time. In *7th International Conference on Formal Structures for Computation and Deduction (FSCD 2022), August 2-5, Haifa, Israel*, volume 228. LIPIcs, 2022.

[AHS20] Matteo Acclavio, Ross Horne, and Lutz Straßburger. Logic beyond formulas: A proof system on graphs. In Holger Hermanns, Lijun Zhang, Naoki Kobayashi, and Dale Miller, editors, *LICS '20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science, Saarbrücken, Germany, July 8-11, 2020*, pages 38–52. ACM, 2020. Full version with technical appendices available at `https://hal.inria.fr/hal-02560105`. `doi:10.1145/3373718.3394763`.

[AL01]   Arnon Avron and Iddo Lev. Canonical propositional Gentzen-type systems. In Rajeev Goré, Alexander Leitsch, and Tobias Nipkow, editors, *Automated Reasoning*, pages 529–544, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

[AM20]   Matteo Acclavio and Roberto Maieli. Generalized connectives for multiplicative linear logic. In Maribel Fernández and Anca Muscholl, editors, *28th EACSL Annual Conference on Computer Science Logic (CSL 2020)*, volume 152 of *LIPIcs*, pages 6:1–6:16, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. URL: `https://drops.dagstuhl.de/opus/volltexte/2020/11649`, `doi:10.4230/LIPIcs.CSL.2020.6`.

[And92] Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. *Journal of Logic and Computation*, 2(3):297–347, 1992.

[AT17]   Andrea Aler Tubella. *A study of normalisation through subatomic logic*. PhD thesis, University of Bath, 2017.

[ATG18] Andrea Aler Tubella and Alessio Guglielmi. Subatomic proof systems: Splittable systems. *ACM Trans. Comput. Logic*, 19(1), January 2018. `doi:10.1145/3173544`.

[BCST96] Richard Blute, Robin Cockett, Robert Seely, and Todd Trimble. Natural deduction and coherence for weakly distributive categories. *J. of Pure and Applied Algebra*, 113(3):229–296, 1996.

[Bel97] Gianluigi Bellin. Subnets of proof-nets in multiplicative linear logic with mix. *Mathematical Structures in Computer Science*, 7(6):663–669, 1997. `doi:10.1017/S0960129597002326`.

[Ber78] Gérard Berry. Stable models of typed λ-calculi. In Giorgio Ausiello and Corrado Böhm, editors, *Automata, Languages and Programming*, pages 72–89, Berlin, Heidelberg, 1978. Springer.

[BG09]   Paola Bruscoli and Alessio Guglielmi. On the proof complexity of deep inference. *ACM Trans. Comput. Logic*, 10(2), March 2009. `doi:10.1145/1462179.1462186`.

[BG16]   Paola Bruscoli and Alessio Guglielmi. On analyticity in deep inference. note, 2016. URL: `http://cs.bath.ac.uk/ag/p/ADI.pdf`.

[BM90]   Anthony J. Bonner and L. Thorne McCarty. Adding negation-as-failure to intuitionistic logic programming. In *Proceedings of the 1990 North American Conference on Logic Programming*, page 681–703, Cambridge, MA, USA, 1990. MIT Press.

[Bru02] Paola Bruscoli. A purely logical account of sequentiality in proof search. In Peter J. Stuckey, editor, *Logic Programming*, pages 302–316, Berlin, Heidelberg, 2002. Springer. `doi:10.1007/3-540-45619-8\_21`.

[Brü03] Kai Brünnler. *Deep Inference and Symmetry for Classical Proofs*. PhD thesis, Technische Universität Dresden, 2003.

[BS17]   Paola Bruscoli and Lutz Straßburger. On the length of medial-switch-mix derivations. In Juliette Kennedy and Ruy J. G. B. de Queiroz, editors, *Logic, Language, Information, and Computation - 24th International Workshop, WoLLIC 2017, London, UK, July 18-21, 2017, Proceedings*, volume 10388 of *Lecture Notes in Computer Science*, pages 68–79. Springer, 2017. `doi:10.1007/978-3-662-55386-2\_5`.

[BT01]   Kai Brünnler and Alwen Fernanto Tiu. A local system for classical logic. In Robert Nieuwenhuis and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning*, pages 347–361, Berlin, Heidelberg, 2001. Springer. `doi:10.1007/3-540-45653-8\_24`.

[Cal16] Cameron Calk. A graph theoretical extension of boolean logic. Bachelor's thesis, 2016. URL: `http://www.anupamdas.com/graph-bool.pdf`.

[CD08] Bruno Courcelle and Christian Delhommé. The modular decomposition of countable graphs. definition and construction in monadic second-order logic. *Theoretical Computer Science*, 394(1):1–38, 2008. URL: `https://www.sciencedirect.com/science/article/pii/S0304397507008171`, `doi:https://doi.org/10.1016/j.tcs.2007.10.046`.

[CDW20] Cameron Calk, Anupam Das, and Tim Waring. Beyond formulas-as-cographs: an extension of boolean logic to arbitrary graphs, 2020. `arXiv:2004.12941`.

[CGS11] Kaustuv Chaudhuri, Nicolas Guenot, and Lutz Straßburger. The focused calculus of structures. In Marc Bezem, editor, *CSL'11*, volume 12 of *LIPIcs*, pages 159–173, Dagstuhl, Germany, 2011. Leibniz-Zentrum fuer Informatik. `doi:10.4230/LIPIcs.CSL.2011.159`.

[CR79] Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *J. Symb. Log.*, 44(1):36–50, 1979. `doi:10.2307/2273702`.

[Das19] Anupam Das. Complexity of evaluation and entailment in boolean graph logic. preprint, 2019. URL: `http://www.anupamdas.com/complexity-graph-bool-note.pdf`.

[DP04] Kosta Došen and Zoran Petrić. *Proof-Theoretical Coherence*. KCL Publ., London, 2004.

[DR89] Vincent Danos and Laurent Regnier. The structure of the multiplicatives. *Arch. Math. Log.*, 28(3):181–203, 1989. `doi:10.1007/BF01622878`.

[DR21] Anupam Das and Alex A. Rice. New minimal linear inferences in boolean logic independent of switch and medial. In Naoki Kobayashi, editor, *6th International Conference on Formal Structures for Computation and Deduction, FSCD 2021, July 17-24, 2021, Buenos Aires, Argentina (Virtual Conference)*, volume 195 of *LIPIcs*, pages 14:1–14:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. `doi:10.4230/LIPIcs.FSCD.2021.14`.

[DS15] Anupam Das and Lutz Straßburger. No complete linear term rewriting system for propositional logic. In Maribel Fernández, editor, *26th International Conference on Rewriting Techniques and Applications (RTA 2015)*, volume 36 of *LIPIcs*, pages 127–142, Dagstuhl, Germany, 2015. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. URL: `http://drops.dagstuhl.de/opus/volltexte/2015/5193`, `doi:10.4230/LIPIcs.RTA.2015.127`.

[DS16] Anupam Das and Lutz Straßburger. On linear rewriting systems for boolean logic and some applications to proof theory. *Logical Methods in Computer Science*, 12(4):1–27, 2016. `doi:10.2168/LMCS-12(4:9)2016`.

[DSC16] Yuxin Deng, Robert J. Simmons, and Iliano Cervesato. Relating reasoning methodologies in linear logic and process algebra. *Mathematical Structures in Computer Science*, 26(5):868–906, 2016. `doi:10.1017/S0960129514000413`.

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

[EHR99] A Ehrenfeucht, T Harju, and G Rozenberg. *The Theory of 2-Structures*. WORLD SCIENTIFIC, 1999. URL: `https://www.worldscientific.com/doi/abs/10.1142/4197`, `arXiv:https://www.worldscientific.com/doi/pdf/10.1142/4197`, `doi:10.1142/4197`.

[FR94] Arnaud Fleury and Christian Retoré. The mix rule. *Mathematical Structures in Computer Science*, 4(2):273–285, 1994. `doi:10.1017/S0960129500000451`.

[FRS01] François Fages, Paul Ruet, and Sylvain Soliman. Linear concurrent constraint programming: Operational and phase semantics. *Information and Computation*, 165(1):14–41, 2001. URL: `https://www.sciencedirect.com/science/article/pii/S0890540100930025`, `doi:10.1006/inco.2000.3002`.

[Gab98] Dov M. Gabbay. *Fibring logics*. Oxford Logic Guides. Clarendon Press, 1998.

[Gal67] Tibor Gallai. Transitiv orientierbare Graphen. *Acta Mathematica Academiae Scientiarum Hungarica*, 18(1–2):25–66, 1967.

[Gen35a] Gerhard Gentzen. Untersuchungen über das logische Schließen. I. *Mathematische Zeitschrift*, 39:176–210, 1935.

[Gen35b] Gerhard Gentzen. Untersuchungen über das logische Schließen. II. *Mathematische Zeitschrift*, 39:405–431, 1935.

[GGP10] Alessio Guglielmi, Tom Gundersen, and Michel Parigot. A proof calculus which reduces syntactic bureaucracy. In Christopher Lynch, editor, *Proceedings of the 21st International Conference on Rewriting Techniques and Applications*, volume 6 of *LIPIcs*, pages 135–150, Dagstuhl, Germany, 2010. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. URL: `http://drops.dagstuhl.de/opus/volltexte/2010/2649`, `doi:10.4230/LIPIcs.RTA.2010.135`.

[Gir87a]   Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987. `doi:10.1016/0304-3975(87)90045-4`.

[Gir87b]   Jean-Yves Girard. Multiplicatives. In Gabriele Lolli, editor, *Logic and Computer Science: New Trends and Applications*, pages 11–34. Rosenberg & Sellier, 1987.

[Gir00]   Jean-Yves Girard. On the meaning of logical rules II: multiplicatives and additives. *NATO ASI Series F: Computer and Systems Sciences*, 175:183–212, 2000.

[GLT89]   Jean-Yves Girard, Yves Lafont, and Paul Taylor. *Proofs and Types*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1989.

[GS01]   Alessio Guglielmi and Lutz Straßburger. Non-commutativity and MELL in the calculus of structures. In Laurent Fribourg, editor, *Computer Science Logic*, pages 54–68, Berlin, Heidelberg, 2001. Springer. `doi:10.1007/3-540-44802-0_5`.

[GS02]   Alessio Guglielmi and Lutz Straßburger. A non-commutative extension of MELL. In Matthias Baaz and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning*, pages 231–246, Berlin, Heidelberg, 2002. Springer. `doi:10.1007/3-540-36078-6_16`.

[GS11]   Alessio Guglielmi and Lutz Straßburger. A system of interaction and structure V: the exponentials and splitting. *Mathematical Structures in Computer Science*, 21(3):563–584, 2011. `doi:https://doi.org/10.1017/S096012951100003X`.

[Gug07]   Alessio Guglielmi. A system of interaction and structure. *ACM Transactions on Computational Logic*, 8(1):1–64, 2007. `doi:10.1145/1182613.1182614`.

[HdMP04]   Michel Habib, Fabien de Montgolfier, and Christophe Paul. A simple linear-time modular decomposition algorithm for graphs, using order extension. In Torben Hagerup and Jyrki Katajainen, editors, *Algorithm Theory - SWAT 2004*, pages 187–198, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[Hor15]   R. Horne. The consistency and complexity of multiplicative additive system virtual. *Scientific Annals of Computer Science*, 25(2):245–316, 2015. `doi:10.7561/SACS.2015.2.245`.

[Hor19]   Ross Horne. The sub-additives: A proof theory for probabilistic choice extending linear logic. In Herman Geuvers, editor, *4th International Conference on Formal Structures for Computation and Deduction, FSCD 2019, June 24-30, 2019, Dortmund, Germany*, volume 131 of *LIPIcs*, pages 23:1–23:16, Dagstuhl, Germany, 2019. Leibniz-Zentrum für Informatik. URL: `http://www.dagstuhl.de/dagpub/978-3-95977-107-8`, `doi:10.4230/LIPIcs.FSCD.2019.23`.

[Hor20]   Ross Horne. Session subtyping and multiparty compatibility using circular sequents. In Igor Konnov and Laura Kovács, editors, *31st International Conference on Concurrency Theory (CONCUR 2020)*, volume 171 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 12:1–12:22, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. URL: `https://drops.dagstuhl.de/opus/volltexte/2020/12824`, `doi:10.4230/LIPIcs.CONCUR.2020.12`.

[How80]   William Alvin Howard. The formulae-as-types notion of construction. In J. P. Seldin and J. R. Hindley, editors, *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 479–490. Academic Press, London, 1980.

[HP10]   Michel Habib and Christophe Paul. A survey of the algorithmic aspects of modular decomposition. *Computer Science Review*, 4(1):41–59, 2010. URL: `https://www.sciencedirect.com/science/article/pii/S157401371000002X`, `doi:https://doi.org/10.1016/j.cosrev.2010.01.001`.

[HT19]   Ross Horne and Alwen Tiu. Constructing weak simulations from linear implications for processes with private names. *Mathematical Structures in Computer Science*, 29(8):1275–1308, 2019. `doi:10.1017/S0960129518000452`.

[HTAC19]   Ross Horne, Alwen Tiu, Bogdan Aman, and Gabriel Ciobanu. De Morgan dual nominal quantifiers modelling private names in non-commutative logic. *ACM Trans. Comput. Log.*, 20(4):22:1–22:44, 2019. `doi:10.1145/3325821`.

[Hug06]   Dominic Hughes. Proofs Without Syntax. *Annals of Mathematics*, 164(3):1065–1076, 2006. `doi:10.4007/annals.2006.164.1065`.

[KY93]   Naoki Kobayashi and Akinori Yonezawa. ACL –a concurrent linear logic programming paradigm. In *Proceedings of the 1993 International Symposium on Logic Programming*, ILPS '93, pages 279–294, Cambridge, MA, USA, 1993. MIT Press.

[Laf95]   Yves Lafont. From proof nets to interaction nets. In J.-Y. Girard, Y. Lafont, and L. Regnier, editors, *Advances in Linear Logic*, volume 222 of *London Mathematical Society Lecture Notes*, pages 225–247. Cambridge University Press, 1995.

[Lam61]   Jim Lambek. On the calculus of syntactic types. In R. Jacobson, editor, *Proceedings of the Twelfth Symposium in Applied Mathematics*, volume XII, pages 166–178, 1961.

[LW00]   Kamal Lodaya and Pascal Weil. Series–parallel languages and the bounded-width property. *Theoretical Computer Science*, 237(1):347 – 380, 2000. URL: `http://www.sciencedirect.com/science/article/pii/S0304397500000311`, `doi:https://doi.org/10.1016/S0304-3975(00)00031-1`.

[Mai19]   Roberto Maieli. Non decomposable connectives of linear logic. *Annals of Pure and Applied Logic*, 170(11):102709, 2019. URL: `http://www.sciencedirect.com/science/article/pii/S0168007219300600`, `doi:https://doi.org/10.1016/j.apal.2019.05.006`.

[Mat68]   Benson Mates. Leibniz on possible worlds. In B. Van Rootselaar and J.F. Staal, editors, *Logic, Methodology and Philosophy of Science III*, volume 52 of *Studies in Logic and the Foundations of Mathematics*, pages 507–529. Elsevier, 1968. URL: `http://www.sciencedirect.com/science/article/pii/S0049237X0871214X`, `doi:10.1016/S0049-237X(08)71214-X`.

[Mil93]   Dale Miller. The π-calculus as a theory in linear logic: Preliminary results. In E. Lamma and P. Mello, editors, *Extensions of Logic Programming*, pages 242–264, Berlin, Heidelberg, 1993. Springer. `doi:10.1007/3-540-56454-3_13`.

[MNPS91] Dale Miller, Gopalan Nadathur, Frank Pfenning, and Andre Scedrov. Uniform proofs as a foundation for logic programming. *Annals of Pure and Applied logic*, 51(1-2):125–157, 1991. `doi:10.1016/0168-0072(91)90068-W`.

[Möh89]   Rolf H. Möhring. Computationally tractable classes of ordered sets. In Ivan Rival, editor, *Algorithms and Order*, pages 105–193, Dordrecht, 1989. Springer Netherlands. `doi:10.1007/978-94-009-2639-4_4`.

[MP13]   Dale Miller and Elaine Pimentel. A formal framework for specifying sequent calculus proof systems. *Theoretical Computer Science*, 474:98–116, 2013.

[MR84]   R.H. Möhring and F.J. Radermacher. Substitution decomposition for discrete structures and connections with combinatorial optimization. In R.E. Burkard, R.A. Cuninghame-Green, and U. Zimmermann, editors, *Algebraic and Combinatorial Methods in Operations Research*, volume 95 of *North-Holland Mathematics Studies*, pages 257–355. North-Holland, 1984. URL: `https://www.sciencedirect.com/science/article/pii/S0304020808729669`, `doi:https://doi.org/10.1016/S0304-0208(08)72966-9`.

[MS94]   Ross M. McConnell and Jeremy P. Spinrad. Linear-time modular decomposition and efficient transitive orientation of comparability graphs. In *Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '94, pages 536–545, USA, 1994. Society for Industrial and Applied Mathematics.

[MS14]   Sonia Marin and Lutz Straßburger. Label-free Modular Systems for Classical and Intuitionistic Modal Logics. In *Advances in Modal Logic 10*, Groningen, Netherlands, August 2014.

[NOP17]   Vivek Nigam, Carlos Olarte, and Elaine Pimentel. On subexponentials, focusing and modalities in concurrent systems. *Theoretical Computer Science*, 693:35–58, 2017. URL: `https://www.sciencedirect.com/science/article/pii/S0304397517305212`, `doi:10.1016/j.tcs.2017.06.009`.

[NPW81]   Mogens Nielsen, Gordon Plotkin, and Glynn Winskel. Petri nets, event structures and domains, part I. *Theoretical Computer Science*, 13(1):85 – 108, 1981. URL: `http://www.sciencedirect.com/science/article/pii/0304397581901122`, `doi:https://doi.org/10.1016/0304-3975(81)90112-2`.

[NS22a]   Lê Thành Dũng Nguyên and Lutz Straßburger. BV and Pomset Logic Are Not the Same. In Florin Manea and Alex Simpson, editors, *30th EACSL Annual Conference on Computer Science Logic (CSL 2022)*, volume 216 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 32:1–32:17, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. URL: `https://drops.dagstuhl.de/opus/volltexte/2022/15752`, `doi:10.4230/LIPIcs.CSL.2022.32`.

[NS22b]   Lê Thành Dũng Nguyên and Lutz Straßburger. A system of interaction and structure iii: The complexity of bv and pomset logic, 2022. URL: `https://arxiv.org/abs/2209.07825`, `doi:10.48550/ARXIV.2209.07825`.

[OP17]   Carlos Olarte and Elaine Pimentel. On concurrent behaviors and focusing in linear logic. *Theoretical Computer Science*, 685:46–64, 2017. Logical and Semantic Frameworks with Applications. URL: `https://www.sciencedirect.com/science/article/pii/S0304397516304832`, `doi:10.1016/j.tcs.2016.08.026`.

[Pet77]   Carl Adam Petri. Interpretations of net theory. *ACM Computing Surveys*, 9(3):223–252, 1977.

[Pog08]   Francesca Poggiolesi. A cut-free simple sequent calculus for modal logic S5. *The Review of Symbolic Logic*, 1(1):3–15, 2008. `doi:10.1017/S1755020308080040`.

[Pra86]   Vaughan Pratt. Modeling concurrency with partial orders. *International Journal of Parallel Programming*, 15(1):33–71, 1986. `doi:10.1007/BF01379149`.

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

[Ret96]   Christian Retoré. Perfect matchings and series-parallel graphs: multiplicatives proof nets as r&b-graphs. *Electronic Notes in Theoretical Computer Science*, 3, 1996.

[Ret97]   Christian Retoré. Pomset logic: A non-commutative extension of classical linear logic. In Philippe de Groote and J. Roger Hindley, editors, *Typed Lambda Calculi and Applications*, pages 300–318, Berlin, Heidelberg, 1997. Springer. `doi:10.1007/3-540-62688-3_43`.

[Ret99]   Christian Retoré. Pomset logic as a calculus of directed cographs. In V. M. Abrusci and C. Casadio, editors, *Dynamic Perspectives in Logic and Linguistics*, pages 221–247. Bulzoni, Roma, 1999. Also available as INRIA Rapport de Recherche RR-3714.

[Ret03]   Christian Retoré. Handsome proof-nets: perfect matchings and cographs. *Theoretical Computer Science*, 294(3):473–488, 2003. `doi:10.1016/S0304-3975(01)00175-X`.

[Ret21]   Christian Retoré. Pomset logic. In *Joachim Lambek: The Interplay of Mathematics, Logic, and Linguistics*, pages 299–345. Springer, 2021.

[Sto07]   Phiniki Stouppa. A deep inference system for the modal logic S5. *Studia Logica*, 85(2):199–214, 2007. `doi:10.1007/s11225-007-9028-y`.

[Str02]   Lutz Straßburger. A local system for linear logic. In Matthias Baaz and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning*, pages 388–402, Berlin, Heidelberg, 2002. Springer. `doi:10.1007/3-540-36078-6_26`.

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

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

[Str17]   Lutz Straßburger. Combinatorial Flows and Their Normalisation. In Dale Miller, editor, *2nd International Conference on Formal Structures for Computation and Deduction (FSCD 2017)*, volume 84 of *LIPIcs*, pages 31:1–31:17, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. URL: `http://drops.dagstuhl.de/opus/volltexte/2017/7720`, `doi:10.4230/LIPIcs.FSCD.2017.31`.

[Tiu06]   Alwen Fernanto Tiu. A system of interaction and structure II: The need for deep inference. *Logical Methods in Computer Science*, 2(2):1–24, 2006. `doi:10.2168/LMCS-2(2:4)2006`.

[TM05]   Alwen Tiu and Dale Miller. A proof search specification of the π-calculus. *Electronic Notes in Theoretical Computer Science*, 138(1):79–101, 2005. Proceedings of the Workshop on the Foundations of Global Ubiquitous Computing (FGUC 2004). URL: `https://www.sciencedirect.com/science/article/pii/S1571066105051121`, `doi:https://doi.org/10.1016/j.entcs.2005.05.006`.

[TS19]   Andrea Aler Tubella and Lutz Strassburger. Introduction to Deep Inference. Lecture, August 2019. URL: `https://hal.inria.fr/hal-02390267`.

[War19]   Timothy Waring. A graph theoretic extension of boolean logic. Master's thesis, 2019. URL: `http://anupamdas.com/thesis_tim-waring.pdf`.

Appendix A. Proofs of Splitting and Context Reduction

The following simple lemma is used in several places of the main proof:

Lemma A.1. *Let $C_1[\cdot]_{R_1}, \ldots, C_n[\cdot]_{R_n}$ be contexts. If $\vdash_{\mathsf{GS}} C_i$ for all $i \in \{1, \ldots, n\}$, then $\vdash_{\mathsf{GS}} C_1[C_2[\ldots C_n[\cdot]_{R_n}]_{R_2}]_{R_1}$.*

Proof. We proceed by induction on n. The base case for $n = 1$ is trivial, and the inductive case for $n > 1$ is this derivation:

$$
C_1 \left[\begin{array}{c} \varnothing \\ \mathcal{D}_1 \,\|\, \\ \dfrac{\varnothing}{\begin{array}{c} \mathcal{D}' \,\|\, \\ C_2[\ldots C_n[\cdot]_{R_n}]_{R_2} \end{array}} \end{array} \right]_{R_1} ,
$$

where $\mathcal{D}_n$ is the derivation for C_n and $\mathcal{D}'$ exists by induction hypothesis. $\qquad\square$

Let us now restate and prove the main lemmas of Section 6.

Lemma 6.3 (Splitting Prime). *Let G be a graph and $P \neq \mathbin{⅋}$ be a prime graph with $|V_P| = n$, and $M_1, \ldots, M_n$ be non-empty graphs. If $\vdash_{\mathsf{GS}} G \mathbin{⅋} P(\!|M_1, \ldots, M_n|\!)$, then one of the following holds:*

(A) *either there is a context $C[\cdot]_R$ and graphs $K_1, \ldots, K_n$, such that there are derivations*

$$
\begin{array}{ccccc}
\dfrac{C[P^\perp(\!|K_1, \ldots, K_n|\!)]_R}{\begin{array}{c}\mathcal{D}_G \,\|\, \mathsf{GS} \\ G\end{array}} & , & \dfrac{\varnothing}{\begin{array}{c}\mathcal{D}_i \,\|\, \mathsf{GS} \\ K_i \mathbin{⅋} M_i\end{array}} & and & \dfrac{\varnothing}{\begin{array}{c}\mathcal{D}_C \,\|\, \mathsf{GS} \\ C\end{array}}
\end{array}
$$

for all $i \in \{1, \ldots, n\}$,

(B) *or there is a context $C[\cdot]_R$ and graphs K_X and K_Y, such that there are derivations*

$$
\begin{array}{ccccc}
\dfrac{C[K_X \mathbin{⅋} K_Y]_R}{\begin{array}{c}\mathcal{D}_G \,\|\, \mathsf{GS} \\ G\end{array}} & , \dfrac{\varnothing}{\begin{array}{c}\mathcal{D}_X \,\|\, \mathsf{GS} \\ K_X \mathbin{⅋} M_i\end{array}} , & \dfrac{\varnothing}{\begin{array}{c}\mathcal{D}_Y \,\|\, \mathsf{GS} \\ K_Y \mathbin{⅋} P(\!|M_1, \ldots, M_{i-1}, \varnothing, M_{i+1}, \ldots, M_n|\!)\end{array}} & and & \dfrac{\varnothing}{\begin{array}{c}\mathcal{D}_C \,\|\, \mathsf{GS} \\ C\end{array}}
\end{array}
$$

for some $i \in \{1, \ldots, n\}$.

Proof. We prove the two cases simultaneously, by induction on the size of the graph $G \mathbin{⅋} P(\!|M_1, \ldots, M_n|\!)$, as defined in Definition 5.13. Let $\mathcal{D}$ be the given derivation of $G \mathbin{⅋} P(\!|M_1, \ldots, M_n|\!)$.

We aim to construct $C[\cdot]_R$, $\mathcal{D}_G$, $\mathcal{D}_C$, and either K_i, $\mathcal{D}_i$ for all $i \in \{1, \ldots, n\}$, or K_X, K_Y, $\mathcal{D}_X$, $\mathcal{D}_Y$, satisfying the condition of the statement. We make a case analysis on the bottommost rule instance in the derivation $\mathcal{D}$.

We now proceed, systematically exhausting the cases (a) to (e) described in Section 6. To avoid redundancy in the proof, we assume that whenever we define the context of the shape $C = C_n[\ldots C_2[C_1[\cdot]_{R_1}]_{R_2} \ldots]_{R_n}$ using some derivable contexts $C_1[\cdot]_{R_1}, \ldots, C_n[\cdot]_{R_n}$, then we have a derivation of C defined by Lemma A.1.

(a) If rule r acts inside G or any M_i with $i \in \{1, \ldots, n\}$, then the derivation $\mathcal{D}$ is of shape

$$
\begin{array}{ccc}
\dfrac{\begin{array}{c}\varnothing \\ \mathcal{D}' \,\|\, \mathsf{GS}\end{array}}{\mathsf{r}\,\dfrac{G'}{G} \mathbin{⅋} P(\!|M_1, \ldots, M_n|\!)} & or & \dfrac{\begin{array}{c}\varnothing \\ \mathcal{D}' \,\|\, \mathsf{GS}\end{array}}{G \mathbin{⅋} P\left(\!\Big|M_1, \ldots, M_{i-1}, \mathsf{r}\,\dfrac{M_i'}{M_i}, M_{i+1}, \ldots, M_n\Big|\!\right)}
\end{array}
$$

for some $1 \le i \le n$, and the size of the conclusion of $\mathcal{D}'$ is smaller than the one of $\mathcal{D}$. We apply the induction hypothesis and conclude immediately by adding the corresponding application of r to $\mathcal{D}_G$ or $\mathcal{D}_i$ respectively.

We have a special case for $\mathsf{r} = \mathsf{ai}{\downarrow}$ where $M_i' = \varnothing$ and

$$
G \parr P\left(\!\left|M_1, \ldots, M_{i-1}, \;\mathsf{ai}{\downarrow}\frac{\varnothing}{a^\perp \parr a}, M_{i+1}, \ldots, M_n\right|\!\right)
$$
with $\dfrac{\varnothing}{\mathcal{D}' \parallel \mathsf{GS}}$ above.

This is a base case of the induction, and it is of the form of case (B) where $C[\cdot]_R = K_X = \varnothing$, and $K_Y = G$.

(b) If $G = G' \parr G''$ with $G' \ne \varnothing$ and $\mathcal{D}$ is of shape

$$
\mathsf{s.sw}{\downarrow}\frac{\begin{array}{c}\varnothing \\ \mathcal{D}' \parallel \mathsf{GS} \\ G'' \parr P(\!|M_1, \ldots, M_n|\!)[G']_{R_P}\end{array}}{G'' \parr G' \parr P(\!|M_1, \ldots, M_n|\!)}
$$

The graph $P(\!|M_1, \ldots, M_n|\!)[G']_{R_P}$ cannot be an atom since G' and M_i are non-empty, or a par of two graphs, due to conditions on $\mathsf{s.sw}{\downarrow}$. Then by Lemma 3.9, $P(\!|M_1, \ldots, M_n|\!)[G']_{R_P}$ is composed via a prime graph. We apply the inductive hypothesis. We have the three following possibilities:

(b.I) In this case G moves inside some M_i. W.l.o.g., we can assume $i = 1$, and $\mathcal{D}$ is of the shape

$$
\mathsf{s.sw}{\downarrow}\frac{\begin{array}{c}\varnothing \\ \mathcal{D}' \parallel \mathsf{GS} \\ G'' \parr P(\!|M_1[G']_S, M_2, \ldots, M_n|\!)\end{array}}{G'' \parr G' \parr P(\!|M_1, \ldots, M_n|\!)}
$$

We apply the induction hypothesis to $\mathcal{D}'$ and get one of the following three sub-cases:

(b.I.A) there is a context $C'[\cdot]_{R'}$ and there are graphs $L_1, K_2, \ldots, K_n$ such that there are the following derivations

$$
\begin{array}{c}C'[P^\perp(\!|L_1, K_2, \ldots, K_n|\!)]_{R'} \\ \mathcal{D}''_G \parallel \mathsf{GS} \\ G''\end{array}, \quad \begin{array}{c}\varnothing \\ \mathcal{D}'_1 \parallel \mathsf{GS} \\ L_1 \parr M_1[G']_S\end{array}, \quad \begin{array}{c}\varnothing \\ \mathcal{D}_i \parallel \mathsf{GS} \\ K_i \parr M_i\end{array} \quad \text{and} \quad \begin{array}{c}\varnothing \\ \mathcal{D}'_C \parallel \mathsf{GS} \\ C'\end{array}
$$

for all $i \in \{2, \ldots, n\}$. We let $K_1 = L_1 \parr G'$ and $C[\cdot]_R = C'[\cdot]_{R'}$. Then we conclude since $\mathcal{D}_G$ and $\mathcal{D}_1$ are the following derivations

$$
\begin{array}{c}\mathsf{s.sw}{\downarrow}\dfrac{C[P^\perp(\!|L_1 \parr G', K_2, \ldots, K_n|\!)]_R}{C[P^\perp(\!|L_1, \ldots, K_2, \ldots, K_n|\!)]_R} \\ \mathcal{D}''_G \parallel \\ G''\end{array} \parr G' \quad \text{and} \quad \begin{array}{c}\varnothing \\ \mathcal{D}_1 \parallel \\ L_1 \parr M_1[G']_S \\ \mathsf{s.sw}{\downarrow}\dfrac{}{L_1 \parr G' \parr M_1}\end{array}
$$

(b.I.B) there is a context $C[\cdot]_R$ and there are two graphs L_X and K_Y such that there are the following derivations

$$
\begin{array}{c}
C[L_X \mathbin{⅋} K_Y]_R \\
\mathcal{D}'_G \,\|\, \mathsf{GS} \\
G''
\end{array}
\quad,\quad
\begin{array}{c}
\varnothing \\
\mathcal{D}'_X \,\|\, \mathsf{GS} \\
L_X \mathbin{⅋} M_1[G']_S
\end{array}
\quad,\quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_Y \,\|\, \mathsf{GS} \\
K_Y \mathbin{⅋} P(\varnothing, M_2, \ldots, M_n)
\end{array}
\quad\text{and}\quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_C \,\|\, \mathsf{GS} \\
C
\end{array}
$$

We let $K_X = L_X \mathbin{⅋} G'$. Then we conclude since $\mathcal{D}_G$ and $\mathcal{D}_X$ are the following derivations

$$
\begin{array}{c}
\text{s.sw}\downarrow \dfrac{C[L_X \mathbin{⅋} G' \mathbin{⅋} K_Y]_R}{\begin{array}{c} C[L_X \mathbin{⅋} K_Y]_R \\ \mathcal{D}'_G \,\| \\ G'' \end{array} \mathbin{⅋} G'}
\end{array}
\quad\text{and}\quad
\begin{array}{c}
\varnothing \\
\mathcal{D}'_X \,\| \\
\text{s.sw}\downarrow \dfrac{L_X \mathbin{⅋} M_1[G']_S}{L_X \mathbin{⅋} G' \mathbin{⅋} M_1}
\end{array}
$$

(b.I.C) there is a context $C[\cdot]_R$ and there are graphs K_X and L_Y such that

$$
\begin{array}{c}
C[K_X \mathbin{⅋} L_Y]_R \\
\mathcal{D}'_G \,\|\, \mathsf{GS} \\
G''
\end{array}
\quad,\quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_X \,\|\, \mathsf{GS} \\
K_X \mathbin{⅋} M_i
\end{array}
\quad,\quad
\begin{array}{c}
\varnothing \\
\mathcal{D}'_Y \,\|\, \mathsf{GS} \\
L_Y \mathbin{⅋} P(M_1[G'], M_1, \ldots, M_{i-1}, \varnothing, M_{i+1}, \ldots, M_n)
\end{array}
\quad\text{and}\quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_C \,\|\, \mathsf{GS} \\
C
\end{array}
$$

for some $i \in \{2, \ldots, n\}$. We let $K_Y = G' \mathbin{⅋} L_Y$. Then we conclude since $\mathcal{D}_G$ and $\mathcal{D}_Y$ are the following derivations

$$
\begin{array}{c}
\text{s.sw}\downarrow \dfrac{C[L_X \mathbin{⅋} G' \mathbin{⅋} K_Y]_R}{\begin{array}{c} C[L_X \mathbin{⅋} K_Y]_R \\ \mathcal{D}'_G \,\| \\ G'' \end{array} \mathbin{⅋} G'}
\end{array}
\quad\text{and}\quad
\begin{array}{c}
\varnothing \\
\mathcal{D}'_Y \,\| \\
\text{s.sw}\downarrow \dfrac{L_Y \mathbin{⅋} P(M_1, \ldots, M_{i-1}, \varnothing, M_{i+1}, \ldots, M_j[G'], \ldots, M_n)}{G' \mathbin{⅋} L_Y \mathbin{⅋} P(M_1, \ldots, M_{i-1}, \varnothing, M_{i+1}, \ldots, M_n)}
\end{array}
\;.
$$

(b.II) There is a special case when the prime graph P is $\otimes$ induced by graph isomorphism, which has the effect of making tensor associative. Assume $P = \otimes$ and $\mathcal{D}$ is of the following shape, where $M_1 = A$ and $M_2 = B$.

$$
\begin{array}{c}
\varnothing \\
\mathcal{D}' \,\|\, \mathsf{GS} \\
G'' \mathbin{⅋} (A \otimes B)[G']_S \\
\text{s.sw}\downarrow \overline{\phantom{G'' \mathbin{⅋} G' \mathbin{⅋} (A \otimes B)}} \\
G'' \mathbin{⅋} G' \mathbin{⅋} (A \otimes B)
\end{array}
$$

with $(A \otimes B)[G']_S \simeq A'' \otimes (A' \otimes B)[G']_{S'}$ for some $S' \subseteq S$, where $A \simeq A'' \otimes A'$ and $A'' \neq \varnothing \neq A'$.

Notice that this is neither the case that G' moves entirely inside A or B (hence Case (b.I) cannot be applied), nor is it the case that G' is entirely a module outside the prime graph $\otimes$ connecting A'' and $(A' \otimes B)[G']_{S'}$, (in which we could move forwards to Case (b.III)). Observe furthermore that such a situation can never occur when P is not $\otimes$.

Since $\|G'' \mathbin{⅋} (A \otimes B)[G']_S\| < \|G'' \mathbin{⅋} G' \mathbin{⅋} (A \otimes B)\|$ we can apply the induction hypothesis in the form of Lemma 6.1. Then there is a context C' and there

are graphs K_A'' and K_Y' such that there are derivations

$$\begin{array}{c} C'[K_A'' \,\bindnasrepma\, K_Y]_{R_1} \\ \mathcal{D}_G'' \,\|\, \mathsf{GS} \\ G'' \end{array} \quad , \quad \begin{array}{c} \varnothing \\ \mathcal{D}_A'' \,\|\, \mathsf{GS} \\ K_A'' \,\bindnasrepma\, A'' \end{array} \quad , \quad \begin{array}{c} \varnothing \\ \mathcal{D}_Y \,\|\, \mathsf{GS} \\ K_Y \,\bindnasrepma\, (A' \otimes B)[G']_{S'} \end{array} \quad \text{and} \quad \begin{array}{c} \varnothing \\ \mathcal{D}_C' \,\|\, \mathsf{GS} \\ C' \end{array} \;.$$

From $\mathcal{D}_Y$ we get that $\vdash_{\mathsf{GS}} K_Y \,\bindnasrepma\, G' \,\bindnasrepma\, (A' \otimes B)$ (via the rule s.sw↓).
It is important to observe at this point that we have the following inequality.

$$\|K_Y \,\bindnasrepma\, G' \,\bindnasrepma\, (A' \otimes B)\| < \|G \,\bindnasrepma\, (A \otimes B)\|$$

This is because, since A'' is non-empty and hence $\|A'\| < \|A\|$ since A' has strictly less vertices, and also $\|K_Y \bindnasrepma G'\| \leq \|G\|$, by Observation 5.14. Therefore, to the proof of $\vdash_{\mathsf{GS}} K_Y \,\bindnasrepma\, G' \,\bindnasrepma\, (A' \otimes B)$ we can apply the induction hypothesis to get a context $C''[\cdot]_{R''}$ and K_A' and K_B such that there are derivations as follows.

$$\begin{array}{c} C''[K_A \,\bindnasrepma\, K_B]_{R''} \\ \mathcal{D}_G' \,\|\, \mathsf{GS} \\ K_Y \,\bindnasrepma\, G' \end{array} \quad , \quad \begin{array}{c} \varnothing \\ \mathcal{D}_A' \,\|\, \mathsf{GS} \\ K_A' \,\bindnasrepma\, A' \end{array} \quad , \quad \begin{array}{c} \varnothing \\ \mathcal{D}_B \,\|\, \mathsf{GS} \\ K_B \,\bindnasrepma\, B \end{array} \quad \text{and} \quad \begin{array}{c} \varnothing \\ \mathcal{D}_C'' \,\|\, \mathsf{GS} \\ C'' \end{array} \;.$$

We conclude by letting $C[\cdot]_R = C'[C''[\cdot]_{R''}]_{R'}$, $K_A = K_A'' \,\bindnasrepma\, K_A'$, and $\mathcal{D}_G$ and $\mathcal{D}_A$ be the derivations defined as

$$\begin{array}{c} C' \left[\begin{array}{c} \mathsf{s.sw\downarrow} \dfrac{C''[K_A'' \,\bindnasrepma\, K_A' \,\bindnasrepma\, K_B]_{R''}}{C''[K_A' \,\bindnasrepma\, K_B]_{R''}} \\ K_A'' \,\bindnasrepma\, \begin{array}{c} \mathcal{D}_G' \,\| \\ K_Y \,\bindnasrepma\, G' \end{array} \end{array} \right]_{R'} \\ \mathsf{s.sw\downarrow} \dfrac{}{\begin{array}{c} C'[K_A'' \,\bindnasrepma\, K_Y]_{R'} \\ \mathcal{D}_G'' \,\| \\ G'' \end{array} \,\bindnasrepma\, G'} \end{array} \quad \text{and} \quad \begin{array}{c} \varnothing \\ \mathcal{D}_A'' \,\| \\ K_A'' \,\bindnasrepma\, \left(A'' \otimes \begin{array}{c} \varnothing \\ \mathcal{D}_{A'} \,\| \\ K_{A'} \,\bindnasrepma\, A' \end{array} \right) \\ \mathsf{s.sw\downarrow} \dfrac{}{K_A'' \,\bindnasrepma\, K_A' \,\bindnasrepma\, (A'' \otimes A')} \end{array} \;.$$

(b.III) This is the most involved sub-case of the proof, induced when the movement of G' via the s.sw↓ rule results in a larger prime graph, where the G' does not overlap with any existing module and possibly adds edges that break up some existing modules of P into smaller modules in the resulting prime graph. Assume $\mathcal{D}$ is of shape

$$\mathsf{s.sw\downarrow} \dfrac{\begin{array}{c} \varnothing \\ \mathcal{D}' \,\|\, \mathsf{GS} \\ G''' \,\bindnasrepma\, Q(\!|G', N_2, \ldots, N_k|\!) \end{array}}{G''' \,\bindnasrepma\, G' \,\bindnasrepma\, P(\!|M_1, \ldots, M_n|\!)}$$

for a unique prime graph Q (up to permutation of modules) such that $k = |V_Q| > |V_P|$ (hence $k \geq 4$) such that

$$Q(\!|\varnothing, N_2, \ldots, N_k|\!) \simeq P(\!|M_1, \ldots, M_n|\!) \tag{A.1}$$

where for each $i \in \{2, \ldots k\}$ we have that $N_i \neq \varnothing$ and N_i is a module of some M_j where $j \in \{1, \ldots n\}$. We apply the induction hypothesis to $G'' \,\mathbin{\gamma\mkern-10mu\gamma}\, Q(\!|G', N_2, \ldots, N_k|\!)$ and get one of the following three sub-cases:

(b.III.A) there is a context $C'[\cdot]_{R'}$ and $K'_1, \ldots, K'_k$ such that there are derivations

$$\begin{array}{c} C'[Q^\perp(\!|K'_1, \ldots, K'_k|\!)]_{R'} \\ \mathcal{D}''_G \,\|\, \mathsf{GS} \\ G'' \end{array} \quad , \quad \begin{array}{c} \varnothing \\ \mathcal{D}'_1 \,\|\, \mathsf{GS} \\ K'_1 \,\mathbin{\gamma\mkern-10mu\gamma}\, G' \end{array} \quad , \quad \begin{array}{c} \varnothing \\ \mathcal{D}'_i \,\|\, \mathsf{GS} \\ K'_i \,\mathbin{\gamma\mkern-10mu\gamma}\, N_i \end{array} \quad \text{and} \quad \begin{array}{c} \varnothing \\ \mathcal{D}'_C \,\|\, \mathsf{GS} \\ C' \end{array}$$

for all $i \in \{2, \ldots, k\}$. Since the graphs $N_2, \ldots, N_n$ are non-empty by hypothesis, we can apply Lemma 5.1 and obtain a derivation

$$\begin{array}{c} \left(\begin{array}{ccc} \begin{array}{c} \varnothing \\ \mathcal{D}'_2 \,\| \\ K'_2 \,\mathbin{\gamma\mkern-10mu\gamma}\, N_2 \end{array} & \otimes \cdots \otimes & \begin{array}{c} \varnothing \\ \mathcal{D}'_k \,\| \\ K'_k \,\mathbin{\gamma\mkern-10mu\gamma}\, N_k \end{array} \end{array} \right) \\ \| \, \text{Lemma 5.1} \\ Q^\perp(\!|\varnothing, K'_2, \ldots, K'_k|\!) \,\mathbin{\gamma\mkern-10mu\gamma}\, Q(\!|\varnothing, N_2, \ldots, N_k|\!) \end{array}$$

By (A.1), such a derivation is a derivation of $Q^\perp(\!|\varnothing, K_2, \ldots, K_k|\!) \,\mathbin{\gamma\mkern-10mu\gamma}\, P(\!|M_1, \ldots, M_n|\!)$ on which we can now apply the inductive hypothesis. This gives the two following cases

- either there is a context $C''[\cdot]_{R''}$ and graphs $K_1, \ldots, K_k$ such that there are derivations

$$\begin{array}{c} C''[P^\perp(\!|K_1, \ldots, K_k|\!)]_{R''} \\ \mathcal{D}_Q \,\|\, \mathsf{GS} \\ Q^\perp(\!|\varnothing, K'_2, \ldots, K'_k|\!) \end{array} \quad , \quad \begin{array}{c} \varnothing \\ \mathcal{D}_i \,\|\, \mathsf{GS} \\ K_i \,\mathbin{\gamma\mkern-10mu\gamma}\, M_i \end{array} \quad \text{and} \quad \begin{array}{c} \varnothing \\ \mathcal{D}''_C \,\|\, \mathsf{GS} \\ C'' \end{array}$$

  for all $i \in \{1, \ldots, n\}$. Then we conclude by letting $C[\cdot]_R = C'[C''[\cdot]_{R''}]_{R'}$ and $\mathcal{D}_G$ be the derivation

$$\begin{array}{c} C' \left[\begin{array}{c} C''[P^\perp(\!|K_1, \ldots, K_k|\!)]_{R''} \\ \mathcal{D}_Q \,\| \\ Q^\perp \left(\left[\begin{array}{c} \varnothing \\ \mathcal{D}'_1 \,\| \\ K'_1 \,\mathbin{\gamma\mkern-10mu\gamma}\, G' \end{array} \right], K'_2, \ldots, K'_k \right) \end{array} \right]_{R'} \\ \mathsf{s.sw}{\downarrow} \, \rule{6cm}{0.5pt} \\ \begin{array}{c} C'[Q^\perp(\!|K'_1, \ldots, K'_k|\!)]_{R'} \\ \mathcal{D}''_G \,\| \\ G'' \end{array} \,\mathbin{\gamma\mkern-10mu\gamma}\, G' \end{array}$$

- or there is a context $C''[\cdot]_{R''}$ and graphs K_X and K_Y such that, w.l.o.g. there are derivations

$$\begin{array}{c} C''[K_X \,\mathbin{\gamma\mkern-10mu\gamma}\, K_Y]_{R''} \\ \mathcal{D}_Q \,\|\, \mathsf{GS} \\ Q^\perp(\!|\varnothing, K'_2, \ldots, K'_k|\!) \end{array} \,, \quad \begin{array}{c} \varnothing \\ \mathcal{D}_X \,\|\, \mathsf{GS} \\ K_X \,\mathbin{\gamma\mkern-10mu\gamma}\, M_1 \end{array} \,, \quad \begin{array}{c} \varnothing \\ \mathcal{D}_Y \,\|\, \mathsf{GS} \\ K_Y \,\mathbin{\gamma\mkern-10mu\gamma}\, P(\!|\varnothing, M_2, \ldots, M_n|\!) \end{array} \quad \text{and} \quad \begin{array}{c} \varnothing \\ \mathcal{D}''_C \,\|\, \mathsf{GS} \\ C'' \end{array}$$

for an $i \in \{1, \ldots, n\}$. Then we conclude by letting $C[\cdot]_R = C'[\cdot]_{R'}$ and $\mathcal{D}_G$ be the derivation

$$
\text{s.sw}\downarrow \cfrac{C'\left[\begin{array}{c} C'''\left[K_X \,\rotatebox[origin=c]{180}{\&}\, K_Y\right]_{R''} \\ \mathcal{D}_Q \,\| \\ Q^\perp\left(\!\left(\begin{array}{c} \varnothing \\ \mathcal{D}'_1 \,\| \\ K'_1 \,\rotatebox[origin=c]{180}{\&}\, G' \end{array}, K'_2, \ldots, K'_k\right)\!\right) \end{array}\right]_{R'}}{\begin{array}{c} C'[K' \,\rotatebox[origin=c]{180}{\&}\, K_2 \,\rotatebox[origin=c]{180}{\&}\, \cdots \,\rotatebox[origin=c]{180}{\&}\, K_n]_{R'} \\ \mathcal{D}''_G \,\| \qquad\qquad \rotatebox[origin=c]{180}{\&}\, G' \\ G'' \end{array}}
$$

(b.III.B) there is a context $C'[\cdot]_{R'}$ and K'_X and K'_Y such that (after A.1) there are derivations

$$
\begin{array}{c} C'[K'_X \,\rotatebox[origin=c]{180}{\&}\, K'_Y]_R \\ \mathcal{D}''_G \,\|\, \mathsf{GS} \\ G'' \end{array} \qquad,\qquad \begin{array}{c} \varnothing \\ \mathcal{D}'_X \,\|\, \mathsf{GS} \\ K'_X \,\rotatebox[origin=c]{180}{\&}\, G' \end{array}\qquad,\qquad \begin{array}{c} \varnothing \\ \mathcal{D}'_Y \,\|\, \mathsf{GS} \\ K'_Y \,\rotatebox[origin=c]{180}{\&}\, P(\!(M_1, \ldots, M_n)\!) \end{array}\qquad \text{and} \qquad \begin{array}{c} \varnothing \\ \mathcal{D}'_C \,\|\, \mathsf{GS} \\ C' \end{array}
$$

We can now apply inductive hypothesis on $K'_Y \,\rotatebox[origin=c]{180}{\&}\, P(\!(M_1, \ldots, M_n)\!)$ and we have the two following cases:

- there is a context $C''[\cdot]_{R''}$ and graphs $K_1, \ldots, K_n$ such that there are derivations

$$
\begin{array}{c} C''[P^\perp(\!(K_1, \ldots, K_n)\!)]_{R''} \\ \mathcal{D}_K \,\|\, \mathsf{GS} \\ K'_Y \end{array}\qquad,\qquad \begin{array}{c} \varnothing \\ \mathcal{D}_i \,\|\, \mathsf{GS} \\ K_i \,\rotatebox[origin=c]{180}{\&}\, M_i \end{array}\qquad \text{and} \qquad \begin{array}{c} \varnothing \\ \mathcal{D}''_C \,\|\, \mathsf{GS} \\ C'' \end{array}
$$

In this case we can conclude similarly to the first case of (b.III.A), that is, by letting $C[\cdot]_R = C'[C''[\cdot]_{R''}]_{R'}$ and $\mathcal{D}_G$ be the derivation

$$
\text{s.sw}\downarrow \cfrac{C'\left[\begin{array}{cc} C''[P^\perp(\!(K_1, \ldots, K_n)\!)]_{R''} & \varnothing \\ \mathcal{D}_K \,\| & \rotatebox[origin=c]{180}{\&}\, \mathcal{D}'_X \,\| \\ K'_Y & K'_X \,\rotatebox[origin=c]{180}{\&}\, G' \end{array}\right]_{R'}}{\begin{array}{c} C'[K'_X \,\rotatebox[origin=c]{180}{\&}\, K'_Y]_{R'} \\ \mathcal{D}''_G \,\| \qquad\qquad \rotatebox[origin=c]{180}{\&}\, G' \\ G'' \end{array}}
$$

- or there is a context $C''[\cdot]_{R''}$ and graphs K_X and K_Y such that, w.l.o.g. there are derivations

$$
\begin{array}{c} C''[K_X \,\rotatebox[origin=c]{180}{\&}\, K_Y]_{R''} \\ \mathcal{D}_K \,\|\, \mathsf{GS} \\ K'_Y \end{array}\qquad,\qquad \begin{array}{c} \varnothing \\ \mathcal{D}_X \,\|\, \mathsf{GS} \\ K_X \,\rotatebox[origin=c]{180}{\&}\, M_1 \end{array}\qquad,\qquad \begin{array}{c} \varnothing \\ \mathcal{D}_Y \,\|\, \mathsf{GS} \\ K_Y \,\rotatebox[origin=c]{180}{\&}\, P(\!(\varnothing, M_2, \ldots, M_n)\!) \end{array}\qquad \text{and} \qquad \begin{array}{c} \varnothing \\ \mathcal{D}''_C \,\|\, \mathsf{GS} \\ C'' \end{array}
$$

we conclude by letting $C[\cdot]_R = C'[C''[\cdot]_{R''}]_{R'}$ and $\mathcal{D}_G$ be the derivation

$$
\text{s.sw}\downarrow \cfrac{C'\left[\begin{array}{cc} \begin{array}{c} C''[K_X \mathbin{\bindnasrepma} K_Y]_{R''} \\ \mathcal{D}_K \,\Vert \\ K'_Y \end{array} & \mathbin{\bindnasrepma} \begin{array}{c} \varnothing \\ \mathcal{D}'_X \,\Vert \\ K'_X \mathbin{\bindnasrepma} G' \end{array} \end{array}\right]_{R'}}{\begin{array}{cc} \begin{array}{c} C'[K'_X \mathbin{\bindnasrepma} K'_Y]_{R'} \\ \mathcal{D}''_G \,\Vert \\ G'' \end{array} & \mathbin{\bindnasrepma} G' \end{array}}
$$

(b.III.C) There is a context $C'[\cdot]_{R'}$ and there are graphs K'_X and K'_Y such that for have the following derivations for some $\ell \in \{2, \dots k\}$.

$$
\begin{array}{c} C'[K'_X \mathbin{\bindnasrepma} K'_Y]_{R'} \\ \mathcal{D}''_G \,\Vert\, \text{GS} \\ G'' \end{array} \;,\; \begin{array}{c} \varnothing \\ \mathcal{D}'_X \,\Vert\, \text{GS} \\ K'_X \mathbin{\bindnasrepma} N_\ell \end{array} \;,\; \begin{array}{c} \varnothing \\ \mathcal{D}'_Y \,\Vert\, \text{GS} \\ K'_Y \mathbin{\bindnasrepma} Q(\!|G', N_2, \dots, N_{\ell-1}, \varnothing, N_{\ell+1}, \dots N_k|\!) \end{array} \quad \text{and} \quad \begin{array}{c} \varnothing \\ \mathcal{D}'_C \,\Vert\, \text{GS} \\ C' \end{array}
$$

There are two cases to consider: either $N_\ell = M_m$ for some $m \in \{1, \dots, k\}$ or we have $M_m = M'[N_\ell]_{R_m}$ for some non-empty M'.
We consider first the former case where $N_\ell = M_m$ for some $m \in \{1, \dots, k\}$, the derivation $\mathcal{D}_Y$ is defined, recalling that by (A.1) we have $Q(\!|\varnothing, N_2, \dots, N_k|\!) \simeq P(\!|M_1, \dots, M_n|\!)$ and hence
$Q(\!|\varnothing, N_2, \dots, N_{\ell-1}, \varnothing, N_{\ell+1}, \dots N_k|\!) \simeq P(\!|M_1, \dots, M_{m-1}, \varnothing, M_{m+1}, \dots M_n|\!)$, i.e.,

$$
\text{s.sw}\downarrow \cfrac{\begin{array}{c} \varnothing \\ \mathcal{D}'_Y \,\Vert \\ K'_Y \mathbin{\bindnasrepma} Q(\!|G', N_2, \dots, N_{\ell-1}, \varnothing, N_{\ell+1}, \dots N_k|\!) \end{array}}{K'_Y \mathbin{\bindnasrepma} G' \mathbin{\bindnasrepma} P(\!|M_1, \dots, M_{m-1}, \varnothing, M_{m+1}, \dots M_n|\!)}
$$

We can conclude almost immediately by letting $C[\cdot]_R = C'[\cdot]_{R'}$, $K_Y = K'_Y \mathbin{\bindnasrepma} G'$, $K_X = K'_X$, $\mathcal{D}_X = \mathcal{D}'_X$, $\mathcal{D}_Y = \mathcal{D}'_Y$, and $\mathcal{D}_G$ be the following derivation.

$$
\text{s.sw}\downarrow \cfrac{C'[K_Y \mathbin{\bindnasrepma} K_X]_{R'}}{\begin{array}{cc} \begin{array}{c} C'[K'_X \mathbin{\bindnasrepma} K'_Y]_{R'} \\ \mathcal{D}''_G \,\Vert \\ G'' \end{array} & \mathbin{\bindnasrepma} G' \end{array}} \quad .
$$

Otherwise, we pursue the case where $M_m = M'[N_\ell]_{R_m}$ for some non-empty M', which is more involved than the case above. By (A.1) we have

$$
Q(\!|\varnothing, N_2, \dots, N_{\ell-1}, \varnothing, N_{\ell+1}, \dots N_k|\!) \simeq P(\!|M_1, \dots, M_{m-1}, M', M_{m+1}, \dots M_n|\!)
$$

Hence, we have a proof of $K'_Y \, \invamp \, G' \, \invamp \, P (\!| M_1, \ldots, M_{m-1}, M', M_{m+1}, \ldots M_n |\!)$ obtained by applying the rule s.sw↓ to the conclusion of $\mathcal{D}'_Y$ above, as follows.

$$
\text{s.sw↓} \; \frac{\begin{array}{c} \varnothing \\ \mathcal{D}'_Y \, \| \\ K'_Y \, \invamp \, Q (\!| G', N_2, \ldots, N_{\ell-1}, \varnothing, N_{\ell+1}, \ldots N_k |\!) \end{array}}{K'_Y \, \invamp \, G' \, \invamp \, P (\!| M_1, \ldots, M_{m-1}, M', M_{m+1}, \ldots M_n |\!)}
$$

By Observation 5.14 $\|K'_Y \, \invamp \, G'\| \leq \|G\|$ and also $N_\ell \neq \varnothing$, hence we have the following inequality.

$$
\|K'_Y \, \invamp \, G' \, \invamp \, P (\!| M_1, \ldots, M_{m-1}, M', M_{m+1}, \ldots M_n |\!)\| < \|G \, \invamp \, P (\!| M_1, \ldots, M_n |\!)\|
$$

Therefore, we can apply the induction hypothesis to the above mentioned proof of $K_Y \, \invamp \, G' \, \invamp \, P (\!| M_1, \ldots, M_{m-1}, M', M_{m+1}, \ldots M_n |\!)$ to obtain one of the following three sub-cases:

(b.III.C.A) There is a context C'' and there are graphs K_1, ..., K_{m-1}, L, K_{m+1}, ..., K_n such that there are derivations

$$
\begin{array}{c} C''[P^\perp (\!| K_1, \ldots, K_{m-1}, L, K_{m+1}, \ldots, K_n |\!)]_{R''} \\ \mathcal{D}'''_G \, \| \, \text{GS} \\ K'_Y \, \invamp \, G' \end{array}
\;,\;
\begin{array}{c} \varnothing \\ \mathcal{D}'_m \, \| \, \text{GS} \\ L \, \invamp \, M' \end{array}
\;,\;
\begin{array}{c} \varnothing \\ \mathcal{D}_i \, \| \, \text{GS} \\ K_i \, \invamp \, M_i \end{array}
\;\; \text{and} \;\;
\begin{array}{c} \varnothing \\ \mathcal{D}''_C \, \| \, \text{GS} \\ C'' \end{array}
$$

for $i \in \{1, \ldots, n\}$ such that $i \neq m$.

To conclude we let $C[\cdot]_R = C'[C''[\cdot]_{R''}]_{R'}$ (see Lemma A.1), and $K_m = K'_X \, \invamp \, L$
Then let $\mathcal{D}_G$ be defined as

$$
\text{s.sw↓} \; \frac{C'\left[\text{s.sw↓} \frac{C''[P^\perp (\!| K_1, \ldots, K_{m-1}, K'_X \, \invamp \, L, K_{m+1}, \ldots, K_n |\!)]_{R''}}{K'_X \, \invamp \, \begin{array}{c} C''[P^\perp (\!| K_1, \ldots, K_{m-1}, L, K_{m+1}, \ldots, K_n |\!)]_{R''} \\ \mathcal{D}'''_G \, \| \\ K'_Y \, \invamp \, G' \end{array}} \right]_{R'}}{\begin{array}{c} C'[K'_X \, \invamp \, K'_Y]_{R'} \\ \mathcal{D}''_G \, \| \\ G'' \end{array} \, \invamp \, G'}
$$

and $\mathcal{D}_m$ be defined as

$$
\text{s.sw↓} \; \frac{\begin{array}{c} \varnothing \\ \mathcal{D}'_m \, \| \\ L \, \invamp \, M' \left[\begin{array}{c} \varnothing \\ \mathcal{D}'_X \, \| \\ K'_X \, \invamp \, N_\ell \end{array} \right]_{R_m} \end{array}}{K'_X \, \invamp \, L \, \invamp \, M'[N_\ell]_{R_m}}
$$

(b.III.C.B) there is a context $C''[\cdot]_{R''}$ and there are graphs K_X and K_Y'' such that there are derivations

$$
\begin{array}{c}
C''[K_X \mathbin{\rotatebox[origin=c]{180}{\&}} K_Y'']_{R''} \\
\mathcal{D}_G''' \, \Big\| \, \mathsf{GS} \\
K_Y' \mathbin{\rotatebox[origin=c]{180}{\&}} G'
\end{array}
\quad , \quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_X \, \Big\| \, \mathsf{GS} \\
K_X \mathbin{\rotatebox[origin=c]{180}{\&}} M_1
\end{array}
\quad , \quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_Y'' \, \Big\| \, \mathsf{GS} \\
K_Y'' \mathbin{\rotatebox[origin=c]{180}{\&}} P(\!|\varnothing, M_2, \ldots, M_{m-1}, M', M_{m+1}, \ldots M_n|\!)
\end{array}
\quad \text{and} \quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_C'' \, \Big\| \, \mathsf{GS} \\
C''
\end{array}
$$

We conclude by setting $K_Y = K_X' \mathbin{\rotatebox[origin=c]{180}{\&}} K_Y''$ and $C[\cdot]_R = C'[C''[\cdot]_{R''}]_{R'}$, where the derivations $\mathcal{D}_G$ is defined as

$$
\mathsf{s.sw}\downarrow \cfrac{
C'\left[K_X' \mathbin{\rotatebox[origin=c]{180}{\&}} \begin{array}{c} K_X \mathbin{\rotatebox[origin=c]{180}{\&}} K_Y'' \\ \mathcal{D}_G''' \, \big\| \\ K_Y' \mathbin{\rotatebox[origin=c]{180}{\&}} G' \end{array} \right]_{R'}
}{
\begin{array}{c}
C'[K_X' \mathbin{\rotatebox[origin=c]{180}{\&}} K_Y']_{R'} \\
\mathcal{D}_G'' \, \big\| \\
G''
\end{array} \mathbin{\rotatebox[origin=c]{180}{\&}} G'
}
$$

and the derivation $\mathcal{D}_Y$ is defined as (recall that $M'[N_\ell]_{R_m} \simeq M_m$):

$$
\mathsf{s.sw}\downarrow \cfrac{
K_Y'' \mathbin{\rotatebox[origin=c]{180}{\&}} P\left(\!\left| \varnothing, M_2 \ldots, M_{m-1}, M' \begin{array}{c} \varnothing \\ \mathcal{D}_Y'' \, \big\| \end{array} \left[\begin{array}{c} \varnothing \\ \mathcal{D}_X' \, \big\| \\ K_X' \mathbin{\rotatebox[origin=c]{180}{\&}} N_\ell \end{array} \right]_{R_m}, M_{m+1}, \ldots M_n \right|\!\right)
}{
K_X' \mathbin{\rotatebox[origin=c]{180}{\&}} K_Y'' \mathbin{\rotatebox[origin=c]{180}{\&}} P(\!|\varnothing, M_2, \ldots, M_{m-1}, M'[N_\ell]_{R_m}, M_{m+1}, \ldots M_n|\!)
}
$$

(b.III.C.C) there is a context C' and there are graphs K_X'' and K_Y such that, there are derivations

$$
\begin{array}{c}
C''[K_X'' \mathbin{\rotatebox[origin=c]{180}{\&}} K_Y]_{R''} \\
\mathcal{D}_G''' \, \big\| \\
K_Y' \mathbin{\rotatebox[origin=c]{180}{\&}} G'
\end{array}
\quad , \quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_X'' \, \Big\| \, \mathsf{GS} \\
K_X'' \mathbin{\rotatebox[origin=c]{180}{\&}} M'
\end{array}
\quad , \quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_Y \, \Big\| \, \mathsf{GS} \\
K_Y \mathbin{\rotatebox[origin=c]{180}{\&}} P(\!|M_1, M_2, \ldots, M_{m-1}, \varnothing, M_{m+1}, \ldots M_n|\!)
\end{array}
\quad \text{and} \quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_C'' \, \Big\| \, \mathsf{GS} \\
C''
\end{array}
$$

We conclude by letting $K_X = K_X' \mathbin{\rotatebox[origin=c]{180}{\&}} K_X''$ and $C[\cdot]_R = C'[C''[\cdot]_{R''}]_{R'}$, where the derivations $\mathcal{D}_G$ and $\mathcal{D}_X$ are defined as follows

$$
\mathsf{s.sw}\downarrow \cfrac{
C'\left[\mathsf{s.sw}\downarrow \cfrac{C'''[K_X' \mathbin{\rotatebox[origin=c]{180}{\&}} K_X'' \mathbin{\rotatebox[origin=c]{180}{\&}} K_Y]_{R''}}{K_X' \mathbin{\rotatebox[origin=c]{180}{\&}} \begin{array}{c} C'''[K_X'' \mathbin{\rotatebox[origin=c]{180}{\&}} K_Y]_{R'} \\ \mathcal{D}_G''' \, \big\| \\ K_Y' \mathbin{\rotatebox[origin=c]{180}{\&}} G' \end{array}} \right]_{R''}
}{
\begin{array}{c}
C'[K_X' \mathbin{\rotatebox[origin=c]{180}{\&}} K_Y']_{R'} \\
\mathcal{D}_G'' \, \big\| \\
G''
\end{array} \mathbin{\rotatebox[origin=c]{180}{\&}} G'
}
\quad \text{and} \quad
\mathsf{s.sw}\downarrow \cfrac{
K_X'' \mathbin{\rotatebox[origin=c]{180}{\&}} M' \left[\begin{array}{c} \varnothing \\ \mathcal{D}_X' \, \big\| \\ K_X' \mathbin{\rotatebox[origin=c]{180}{\&}} N_\ell \end{array} \right]_{R_m}
}{
K_X'' \mathbin{\rotatebox[origin=c]{180}{\&}} K_X' \mathbin{\rotatebox[origin=c]{180}{\&}} M'[N_\ell]_{R_m}
}
$$

(c) This is the case where the s.sw↓-rule is applied and $\mathcal{D}$ is of shape

$$
\text{s.sw↓} \frac{\begin{array}{c} \varnothing \\ \mathcal{D}' \, \| \\ G[P(\!|M_1, \ldots, M_n|\!)]_S \end{array}}{G \,⅋\, P(\!|M_1, \ldots, M_n|\!)}
\tag{A.2}
$$

We apply the induction hypthesis in the form of Lemma 6.6 and get a graph K and a context $C'[\cdot]_R$ such that there are derivations

$$
\begin{array}{c} \varnothing \\ \mathcal{D}^* \, \| \, \text{GS} \\ K \,⅋\, P(\!|M_1, \ldots, M_n|\!) \end{array}
\quad , \quad
\begin{array}{c} C'[K \,⅋\, \mathcal{X}]_R \\ \mathcal{D}_G^{\mathcal{X}} \, \| \, \text{GS} \\ G[\mathcal{X}]_S \end{array}
\quad \text{and} \quad
\begin{array}{c} \varnothing \\ \mathcal{D}'_C \, \| \, \text{GS} \\ C' \end{array}
$$

for any graph $\mathcal{X}$. Since the application of s.sw↓ in (A.2) is not trivial, the context C' cannot be empty, and therefore we have $\|K\| < \|G\|$. Hence we can apply the induction hypothesis to the proof $\mathcal{D}^*$ of $K \,⅋\, P(\!|M_1, \ldots, M_n|\!)$, and obtain one of the following two cases:

- either there is a context $C''[\cdot]_R$ and graphs $K_1, \ldots, K_n$, such that there are derivations

$$
\begin{array}{c} C''[P^\perp(\!|K_1, \ldots, K_n|\!)]_R \\ \mathcal{D}_K \, \| \, \text{GS} \\ K \end{array}
\quad , \quad
\begin{array}{c} \varnothing \\ \mathcal{D}_i \, \| \, \text{GS} \\ K_i \,⅋\, M_i \end{array}
\quad \text{and} \quad
\begin{array}{c} \varnothing \\ \mathcal{D}''_C \, \| \, \text{GS} \\ C'' \end{array}
$$

for all $i \in \{1, \ldots, n\}$. Then we can let $C[\cdot]_R = C'[C''[\cdot]_{R''}]_{R'}$ and the derivation $\mathcal{D}_G$ is:

$$
C'\left[\begin{array}{c} C''[P^\perp(\!|K_1, \ldots, K_n|\!)]_{R''} \\ \mathcal{D}_K \, \| \\ K \end{array}\right]_{R'}
\begin{array}{c} \\ \mathcal{D}_G^\varnothing \, \| \\ G \end{array}
$$

- or there is a context $C''[\cdot]_R$ and graphs K_X and K_Y and derivations

$$
\begin{array}{c} C''[K_X \,⅋\, K_Y]_{R''} \\ \mathcal{D}_K \, \| \, \text{GS} \\ K \end{array}
\;,\;
\begin{array}{c} \varnothing \\ \mathcal{D}_X \, \| \, \text{GS} \\ K_X \,⅋\, M_i \end{array}
\;,\;
\begin{array}{c} \varnothing \\ \mathcal{D}_Y \, \| \, \text{GS} \\ K_Y \,⅋\, P(\!|M_1, \ldots, M_{i-1}, \varnothing, M_{i+1}, \ldots, M_n|\!) \end{array}
\quad \text{and} \quad
\begin{array}{c} \varnothing \\ \mathcal{D}''_C \, \| \, \text{GS} \\ C'' \end{array}
$$

for some $i \in \{1, \ldots, n\}$. Then we let $C[\cdot]_R = C'[C''[\cdot]_{R''}]_{R'}$ and $\mathcal{D}_G$ is

$$
C'\left[\begin{array}{c} C''[K_X \,⅋\, K_Y]_{R''} \\ \mathcal{D}'_K \, \| \\ K \end{array}\right]_{R'}
\begin{array}{c} \\ \mathcal{D}_G^\varnothing \, \| \\ G \end{array}
$$

(d) Here we have $G = G' \mathbin{⅋} P^\perp(\!|N_1, \ldots, N_n|\!)$ and $\mathcal{D}$ is of shape

$$
\mathsf{p}{\downarrow} \frac{
\begin{array}{c}
\varnothing \\
\mathcal{D}' \| \\
G' \mathbin{⅋} ((N_1 \mathbin{⅋} M_1) \otimes \cdots \otimes (N_n \mathbin{⅋} M_n))
\end{array}
}{
G' \mathbin{⅋} P^\perp(\!|N_1, \ldots, N_n|\!) \mathbin{⅋} P(\!|M_1, \ldots, M_n|\!)
} \quad .
$$

We apply the induction hypothesis in the form of Lemma 6.7, which gives us a context $C[\cdot]_R$ and graphs $L_1, \ldots, L_n$ such that

$$
\begin{array}{c}
C[L_1 \mathbin{⅋} \cdots \mathbin{⅋} L_n]_R \\
\mathcal{D}'_G \| \\
G'
\end{array}
\quad , \quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_i \| \\
L_i \mathbin{⅋} N_i \mathbin{⅋} M_i
\end{array}
\quad \text{and} \quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_C \| \\
C
\end{array}
$$

for all $i \in \{1, \ldots, n\}$. We let $K_i = L_i \mathbin{⅋} N_i$. Then there is a derivation $\mathcal{D}_G$ defined as

$$
\mathsf{s.sw}{\downarrow} \frac{
C\left[
\begin{array}{c}
P^\perp(\!|L_1 \mathbin{⅋} N_1, \ldots, L_n \mathbin{⅋} N_n|\!) \\
\| \{\mathsf{s.sw}{\downarrow}\} \\
L_1 \mathbin{⅋} \cdots \mathbin{⅋} L_n \mathbin{⅋} P^\perp(\!|N_1, \ldots, N_n|\!)
\end{array}
\right]_R
}{
\begin{array}{c}
C[L_1 \mathbin{⅋} \cdots \mathbin{⅋} L_n]_R \\
\mathcal{D}'_G \| \\
G'
\end{array}
\mathbin{⅋} P^\perp(\!|N_1, \ldots, N_n|\!)
} \quad .
$$

(e) In this case, the $\mathsf{p}{\downarrow}$ rule is applied to a larger prime graph Q in the context of the principal prime graph P. We have in this case that $\mathcal{D}$ is of shape

$$
\mathsf{p}{\downarrow} \frac{
\begin{array}{c}
\varnothing \\
\mathcal{D}' \| \\
G'' \mathbin{⅋} (N_1 \otimes (N_2 \mathbin{⅋} L_2) \otimes \cdots \otimes (N_k \mathbin{⅋} L_k))
\end{array}
}{
G'' \mathbin{⅋} Q(\!|N_1, \ldots, N_k|\!) \mathbin{⅋} P(\!|M_1, \ldots, M_n|\!)
} \quad .
$$

An essential assumption is that all modules of Q must be non-empty. Hence, in this case we assume $G = G'' \mathbin{⅋} Q(\!|N_1, \ldots, N_k|\!)$ where $N_1, \ldots, N_k$ are non-empty graphs, Q is a prime graph with $k = |V_Q| > |V_P|$ such that w.l.o.g. $P(\!|M_1, \ldots, M_n|\!) \simeq Q^\perp(\!|\varnothing, L_2, \ldots L_k|\!)$ for some possibly empty graphs $L_2, \ldots L_k$. Observe at least one module of the prime connective $Q^\perp$ must be empty, and we let that w.l.o.g. to be the first module, otherwise $P^\perp$ and Q are isomorphic, contradicting $|V_Q| > |V_P|$.

We can apply the induction hypothesis in the form of Lemma 6.7, and we get a context $C'[\cdot]_{R'}$ and graphs $K'_1, \ldots, K'_k$, such that

$$
\begin{array}{c}
C'[K'_1 \mathbin{⅋} \cdots \mathbin{⅋} K'_k]_R \\
\mathcal{D}''_G \| \mathsf{GS} \\
G''
\end{array}
\quad , \quad
\begin{array}{c}
\varnothing \\
\mathcal{D}'_1 \| \mathsf{GS} \\
K'_1 \mathbin{⅋} N_1
\end{array}
\quad , \quad
\begin{array}{c}
\varnothing \\
\mathcal{D}'_j \| \mathsf{GS} \\
K'_j \mathbin{⅋} (N_j \mathbin{⅋} L_j)
\end{array}
\quad \text{and} \quad
\begin{array}{c}
\varnothing \\
\mathcal{D}'_C \| \mathsf{GS} \\
C'
\end{array}
$$

for all $j \in \{2, \ldots, k\}$. Now observe, that there is a graph H (that is not necessarily prime), such that $Q(\!|\varnothing, K'_2 \mathbin{⅋} N_2, \ldots, K'_k \mathbin{⅋} N_k|\!) \simeq H(\!|K'_2 \mathbin{⅋} N_2, \ldots, K'_k \mathbin{⅋} N_k|\!)$ and

$Q^\perp(\!|\varnothing, L_2, \ldots L_k|\!) \simeq H^\perp(\!|L_2, \ldots L_k|\!)$. Hence, since for all $i \in \{2, \ldots, k\}$ we have $N_i \neq \varnothing$, we can apply Lemma 5.1 to construct the following proof.

$$
\begin{pmatrix}
\varnothing \\
\mathcal{D}'_2 \| \\
K'_2 \,\reflectbox{$\invamp$}\, N_2 \,\reflectbox{$\invamp$}\, L_2
\end{pmatrix}
\otimes \ldots \otimes
\begin{pmatrix}
\varnothing \\
\mathcal{D}'_k \| \\
K'_k \,\reflectbox{$\invamp$}\, N_k \,\reflectbox{$\invamp$}\, L_k
\end{pmatrix}
\tag{A.3}
$$
$$
\begin{array}{c}
\mathcal{D}_H \| \text{Lemma 5.1} \\
H(\!|K'_2 \,\reflectbox{$\invamp$}\, N_2, \ldots, K'_k \,\reflectbox{$\invamp$}\, N_k|\!) \,\reflectbox{$\invamp$}\, H^\perp(\!|L_2, \ldots, L_k|\!)
\end{array}
$$

Now, observe that we have $H^\perp(\!|L_2, \ldots, L_k|\!) \simeq P(\!|M_1, \ldots, M_n|\!)$. Furthermore, $\|K'_1 \,\reflectbox{$\invamp$}\, \cdots \,\reflectbox{$\invamp$}\, K'_k\| \leq \|G''\|$, and hence we have $\|H(\!|K'_1 \,\reflectbox{$\invamp$}\, N_1, \ldots, K'_k \,\reflectbox{$\invamp$}\, N_k|\!)\| \leq \|G\|$. Also $N_1 \neq \varnothing$ and N_1 does not appear in the conclusion of the proof (A.3) above; hence we have:

$$\|H(\!|K'_2 \,\reflectbox{$\invamp$}\, N_2, \ldots, K'_k \,\reflectbox{$\invamp$}\, N_k|\!) \,\reflectbox{$\invamp$}\, P(\!|M_1, \ldots, M_n|\!)\| < \|G \,\reflectbox{$\invamp$}\, P(\!|M_1, \ldots, M_n|\!)\|$$

Therefore, we can apply the induction hypothesis to the proof in (A.3), giving us one of the following two cases.

- there is a context $C''[\cdot]_{R''}$ and graphs $K_1, \ldots, K_n$ such that there are derivations

$$
\begin{array}{c}
C''[P^\perp(\!|K_1 \,\reflectbox{$\invamp$}\, \ldots K_n|\!)]_{R''} \\
\mathcal{D}'''_G \| \text{GS} \\
Q(\!|\varnothing, K'_2 \,\reflectbox{$\invamp$}\, N_2, \ldots, K'_k \,\reflectbox{$\invamp$}\, N_k|\!)
\end{array}
\quad , \quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_i \| \text{GS} \\
K_i \,\reflectbox{$\invamp$}\, M_i
\end{array}
\quad \text{and} \quad
\begin{array}{c}
\varnothing \\
\mathcal{D}''_C \| \text{GS} \\
C''
\end{array}
$$

for all $i \in \{1, \ldots, n\}$. We conclude by letting $C[\cdot]_R = C'[C''[\cdot]_{R''}]_{R'}$ and the derivation $\mathcal{D}_G$ be defined as

$$
C'\left[
\begin{array}{c}
C'''[P^\perp(\!|K_1 \,\reflectbox{$\invamp$}\, \ldots K_n|\!)]_{R''} \\
\mathcal{D}'''_G \| \\
Q\left(
\begin{array}{c}
\varnothing \\
\mathcal{D}'_1 \| \\
K'_1 \,\reflectbox{$\invamp$}\, N_1
\end{array},\ K'_2 \,\reflectbox{$\invamp$}\, N_2, \ldots, K'_k \,\reflectbox{$\invamp$}\, N_k
\right) \\
\text{s.sw} \downarrow \ \overline{\qquad K'_1 \,\reflectbox{$\invamp$}\, K'_2 \,\reflectbox{$\invamp$}\, \ldots K'_k \,\reflectbox{$\invamp$}\, Q(\!|N_1, \ldots, N_k|\!) \qquad} _{R'}
\end{array}
\right]
$$
$$
\text{s.sw} \downarrow \ \overline{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}
$$
$$
\begin{array}{c}
C'[K'_1 \,\reflectbox{$\invamp$}\, K'_2 \,\reflectbox{$\invamp$}\, \ldots K'_k]_{R'} \\
\mathcal{D}'' \| \qquad\qquad \,\reflectbox{$\invamp$}\, Q(\!|N_1, \ldots, N_k|\!) \\
G''
\end{array}
$$

- or we have a context $C''[\cdot]_{R''}$ and graphs K_X and K_Y such that, w.l.o.g. there are derivations

$$
\begin{array}{c}
C''[K_X \,\reflectbox{$\invamp$}\, K_Y]_{R''} \\
\mathcal{D}'''_G \| \text{GS} \\
Q(\!|\varnothing, K'_2 \,\reflectbox{$\invamp$}\, N_2, \ldots, K'_k \,\reflectbox{$\invamp$}\, N_k|\!)
\end{array}
, \
\begin{array}{c}
\varnothing \\
\mathcal{D}_X \| \text{GS} \\
K_X \,\reflectbox{$\invamp$}\, M_1
\end{array}
, \
\begin{array}{c}
\varnothing \\
\mathcal{D}_Y \| \text{GS} \\
K_Y \,\reflectbox{$\invamp$}\, P(\!|\varnothing, M_2, \ldots, M_n|\!)
\end{array}
\ \text{and} \
\begin{array}{c}
\varnothing \\
\mathcal{D}''_C \| \text{GS} \\
C''
\end{array}
$$

We conclude by letting $C[\cdot]_R = C'[C''[\cdot]_{R''}]_{R'}$ and the derivation $\mathcal{D}_G$ be defined as

$$
\mathsf{s.sw}\downarrow \frac{
C'\left[\mathsf{s.sw}\downarrow \frac{ Q\left(\left\| \begin{array}{c} \varnothing \\ \mathcal{D}'_1 \| \\ K'_1 \mathbin{\bindnasrepma} N_1 \end{array} \right. , K'_2 \mathbin{\bindnasrepma} N_2, \ldots, K'_k \mathbin{\bindnasrepma} N_k \right) \quad \begin{array}{c} C''[K_X \mathbin{\bindnasrepma} K_Y]_{R''} \\ \mathcal{D}'''_G \| \end{array} }{ K'_1 \mathbin{\bindnasrepma} K'_2 \mathbin{\bindnasrepma} \ldots K'_k \mathbin{\bindnasrepma} Q(\!| N_1, \ldots, N_k |\!) } \right]_{R'}
}{
\begin{array}{c} C'[K'_1 \mathbin{\bindnasrepma} K'_2 \mathbin{\bindnasrepma} \ldots K'_k]_{R'} \\ \mathcal{D}'' \| \\ G'' \end{array} \quad \mathbin{\bindnasrepma} Q(\!| N_1, \ldots, N_k |\!)
}
$$

$\square$

Lemma 6.7 (Splitting Multi-tensor)**.** *Let G be a graph, and $A_1, \ldots, A_n$ be non-empty graphs. If $\vdash_{\mathsf{GS}} G \mathbin{\bindnasrepma} (A_1 \otimes \cdots \otimes A_n)$, then there is a context $C[\cdot]_R$ and graphs $K_1, \ldots, K_n$, such that there are derivations*

$$
\begin{array}{c} C[K_1 \mathbin{\bindnasrepma} \cdots \mathbin{\bindnasrepma} K_n]_R \\ \mathcal{D}_G \| \mathsf{GS} \\ G \end{array} , \quad \begin{array}{c} \varnothing \\ \mathcal{D}_i \| \mathsf{GS} \\ K_i \mathbin{\bindnasrepma} A_i \end{array} \quad and \quad \begin{array}{c} \varnothing \\ \mathcal{D}_C \| \mathsf{GS} \\ C \end{array}
$$

for all $i \in \{1, \ldots, n\}$.

Proof. By induction on n. If $n = 2$, then we conclude by Lemma 6.1. If $n \geq 3$, then $A_1 \otimes \cdots \otimes A_n = A_1 \otimes (A_2 \otimes A_3 \otimes \cdots \otimes A_n)$. Then, by Lemma 6.1, there are a context $C'[\cdot]_{R'}$, and graphs K_1 and K' such that there are derivations

$$
\begin{array}{c} C'[K_1 \mathbin{\bindnasrepma} K_Y]_{R'} \\ \mathcal{D}'_G \| \mathsf{GS} \\ G \end{array} , \quad \begin{array}{c} \varnothing \\ \mathcal{D}_1 \| \mathsf{GS} \\ K_1 \mathbin{\bindnasrepma} M_1 \end{array} , \quad \begin{array}{c} \varnothing \\ \mathcal{D}' \| \mathsf{GS} \\ K' \mathbin{\bindnasrepma} (M_2 \otimes M_3 \otimes \cdots \otimes M_n) \end{array} \quad and \quad \begin{array}{c} \varnothing \\ \mathcal{D}'_C \| \mathsf{GS} \\ C' \end{array} \quad .
$$

By applying the same lemma inductively again to the proof $\mathcal{D}'$ of $K' \mathbin{\bindnasrepma} (A_2 \otimes A_3 \otimes \cdots \otimes A_n)$, we obtain a context $C''[\cdot]_{R''}$ and graphs $K_2, \ldots, K_n$ such that there are derivations

$$
\begin{array}{c} C''[K_2 \mathbin{\bindnasrepma} \cdots \mathbin{\bindnasrepma} K_n]_{R''} \\ \mathcal{D}'_K \| \mathsf{GS} \\ K' \end{array} , \quad \begin{array}{c} \varnothing \\ \mathcal{D}_i \| \mathsf{GS} \\ K_i \mathbin{\bindnasrepma} A_i \end{array} \quad and \quad \begin{array}{c} \varnothing \\ \mathcal{D}''_C \| \mathsf{GS} \\ C'' \end{array}
$$

for all $i \in \{2, \ldots, n\}$. We conclude by letting $C[\cdot]_R = C'[C''[\cdot]_{R''}]_{R'}$ and $\mathcal{D}_G$ be the derivation

$$
C' \left[\begin{array}{c} \text{s.sw}\downarrow \dfrac{C''[K_1 \,\invamp\, \cdots \,\invamp\, K_n]_{R''}}{\begin{array}{c} C''[K_2 \,\invamp\, \cdots \,\invamp\, K_n]_{R''} \\ K_1 \,\invamp\, \quad \mathcal{D}'_K \,\|\, \\ K' \end{array}} \\ \mathcal{D}'_G \,\| \\ G \end{array} \right]_{R'}
$$

$\square$

Lemma 6.6 (Context reduction)**.** *Let A be a graph and $G[\cdot]_S$ be a context, such that $\vdash_{\mathsf{GS}} G[A]_S$. Then there are a context $C[\cdot]_R$ and a graph K, such that there are derivations*

$$
\begin{array}{c} C[K \,\invamp\, \mathcal{X}]_R \\ \mathcal{D}_G \,\|\, \mathsf{GS} \\ G[\mathcal{X}]_S \end{array} \quad and \quad \begin{array}{c} \varnothing \\ \mathcal{D}_A \,\|\, \mathsf{GS} \\ K \,\invamp\, A \end{array} \quad and \quad \begin{array}{c} \varnothing \\ \mathcal{D}_C \,\|\, \mathsf{GS} \\ C \end{array}
$$

for any graph $\mathcal{X}$.

Proof. The case when $A = \varnothing$ is trivial, since we can take $C = G$ and $R = S$ and $K = \varnothing$.

Otherwise, without loss of generality $G[A]_S \simeq G'' \,\invamp\, G'[A]_S$ for a graph $G'[A]_S$ which is neither a par nor empty. The base case is where $G' = \varnothing$ and $S = \varnothing$, in which case we can set $K = G''$ and $C = \varnothing$, and the derivations $\mathcal{D}_C$ and $\mathcal{D}_G$ to be trivial. The derivation $\mathcal{D}_A$ is given by the proof of $G[A]_S$, since under these assumptions $K \,\invamp\, A \simeq G'' \,\invamp\, A \simeq G[A]_S$.

If G' is non-empty, we proceed by induction on the size of $G'[A]_S$ as follows. $G'[A]_S$ must be composed via a prime graph P with $P \neq \invamp$. Then, without loss of generality we can assume that $G[A]_S = G'' \,\invamp\, P\langle\!\langle M_1[A]_{S'}, M_2, \ldots, M_n\rangle\!\rangle$. Applying Lemma 6.3 gives us one of the following three cases:

(A) We have $C'[\cdot]_{R'}$ and $K_1, \ldots, K_n$, such that

$$
\begin{array}{c} C'[P^\perp\langle\!\langle K_1, \ldots, K_n\rangle\!\rangle]_{R'} \\ \mathcal{D}''_G \,\|\, \mathsf{GS} \\ G'' \end{array} \;,\quad \begin{array}{c} \varnothing \\ \mathcal{D}_1 \,\|\, \mathsf{GS} \\ K_1 \,\invamp\, M_1[A]_{S'} \end{array} \;,\quad \begin{array}{c} \varnothing \\ \mathcal{D}_i \,\|\, \mathsf{GS} \\ K_i \,\invamp\, M_i \end{array} \quad and \quad \begin{array}{c} \varnothing \\ \mathcal{D}'_C \,\|\, \mathsf{GS} \\ C' \end{array}
$$

for $2 \leq i \leq n$. We can apply the induction hypothesis to $K_1 \,\invamp\, M_1[A]_{S'}$ and obtain K and $C''[\cdot]_{R''}$, such that

$$
\begin{array}{c} \varnothing \\ \mathcal{D}''_C \,\|\, \mathsf{GS} \\ C'' \end{array} \;,\quad \begin{array}{c} \varnothing \\ \mathcal{D}_A \,\|\, \mathsf{GS} \\ K \,\invamp\, A \end{array} \quad and \quad \begin{array}{c} C''[K \,\invamp\, \mathcal{X}]_{R''} \\ \mathcal{D}'_G \,\|\, \mathsf{GS} \\ K_1 \,\invamp\, M_1[\mathcal{X}]_{S'} \end{array}
$$

for any $\mathcal{X}$. We let $C[\cdot]_R = C'[C''[\cdot]_{R''}]_{R'}$, the derivation $\mathcal{D}_C$ is defined by Lemma A.1, and the derivation $\mathcal{D}_G$ is defined as

$$
C'\left[\begin{array}{c}
\left[\begin{array}{cccccc}
C'''[K \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, \mathcal{X}]_{R''} & & \varnothing & & & \varnothing \\
\mathcal{D}'_G \,\| & \otimes & \mathcal{D}_2 \,\| & \otimes \cdots \otimes & & \mathcal{D}_n \,\| \\
K_1 \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, M_1[\mathcal{X}]_{S'} & & K_2 \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, M_2 & & & K_n \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, M_m
\end{array}\right] \\
\rule{0pt}{1em} \mathsf{p}{\downarrow} \,\overline{} \\
P^\perp(\!|K_1,\ldots,K_n|\!) \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, P(\!|M_1[\mathcal{X}]_{S'}, M_2,\ldots, M_n|\!)
\end{array}\right]_{R'}
$$

$$
\mathsf{s.sw}{\downarrow}\,\overline{}
$$

$$
\begin{array}{c}
C'[P^\perp(\!|K_1,\ldots,K_n|\!)]_{R'} \\
\mathcal{D}''_G \,\| \qquad\qquad \mathbin{\rotatebox[origin=c]{180}{$\&$}}\, P(\!|M_1[\mathcal{X}]_{S'}, M_2,\ldots, M_n|\!) \\
G''
\end{array}
$$

where $G[\mathcal{X}]_S = G'' \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, P(\!|M_1[\mathcal{X}]_{S'}, M_2,\ldots, M_n|\!)$.

(B) We have $C'[\cdot]_{R'}$ and K_X and K_Y, such that

$$
\begin{array}{c}
C'[K_X \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, K_Y]_{R'} \\
\mathcal{D}''_G \,\|\, \mathsf{GS} \\
G''
\end{array}
,\qquad
\begin{array}{c}
\varnothing \\
\mathcal{D}_X \,\|\, \mathsf{GS} \\
K_X \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, M_1[A]_{S'}
\end{array}
,\qquad
\begin{array}{c}
\varnothing \\
\mathcal{D}_Y \,\|\, \mathsf{GS} \\
K_Y \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, P(\!|\varnothing, M_2,\ldots, M_n|\!)
\end{array}
\qquad\text{and}\qquad
\begin{array}{c}
\varnothing \\
\mathcal{D}'_C \,\| \\
C'
\end{array}.
$$

We apply the induction hypothesis to $K_X \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, M_1[A]_{S'}$ and get K and $C''[\cdot]_{R''}$, such that

$$
\begin{array}{c}
\varnothing \\
\mathcal{D}''_C \,\|\, \mathsf{GS} \\
C''
\end{array}
,\qquad
\begin{array}{c}
C''[K \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, \mathcal{X}]_{R''} \\
\mathcal{D}'_G \,\|\, \mathsf{GS} \\
K_X \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, M_1[\mathcal{X}]_{S'}
\end{array}
\qquad\text{and}\qquad
\begin{array}{c}
\varnothing \\
\mathcal{D}_A \,\|\, \mathsf{GS} \\
K \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, A
\end{array}
$$

for any $\mathcal{X}$. We let $C[\cdot]_R = C'[K_Y \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, P(\!|C''[\cdot]_{R''}, M_2,\ldots, M_n|\!)]_{R'}$ and obtain $\mathcal{D}_C$ from Lemma A.1, and $\mathcal{D}_G$ is as follows:

$$
C'\left[\begin{array}{c}
\left[\begin{array}{c}
K_Y \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, P\left(\!\left|\begin{array}{c} C''[K \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, \mathcal{X}]_{R''} \\ \mathcal{D}'_G \,\| \\ K_X \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, M_1[\mathcal{X}]_{S'} \end{array}\right., M_2,\ldots, M_n\right|\!\right) \\
\rule{0pt}{1em}\mathsf{s.sw}{\downarrow}\,\overline{} \\
K_X \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, K_Y \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, P(\!|M_1[\mathcal{X}]_{S'}, M_2,\ldots, M_n|\!)
\end{array}\right]_{R'}
\end{array}\right].
$$

$$
\mathsf{s.sw}{\downarrow}\,\overline{}
$$

$$
\begin{array}{c}
C'[K_X \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, K_Y]_{R'} \\
\mathcal{D}''_G \,\| \qquad\qquad \mathbin{\rotatebox[origin=c]{180}{$\&$}}\, P(\!|M_1[\mathcal{X}]_{S'}, M_2,\ldots, M_n|\!) \\
G''
\end{array}
$$

(C) We have $C'[\cdot]_{R'}$ and K_X and K_Y, such that

$$
\begin{array}{c}
C'[K_X \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, K_Y]_{R'} \\
\mathcal{D}''_G \,\|\, \mathsf{GS} \\
G''
\end{array}
,\quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_X \,\|\, \mathsf{GS} \\
K_X \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, M_2
\end{array}
,\quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_Y \,\|\, \mathsf{GS} \\
K_Y \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, P(\!|M_1[A]_{S'}, \varnothing, M_3,\ldots, M_n|\!)
\end{array}
\quad\text{and}\quad
\begin{array}{c}
\varnothing \\
\mathcal{D}'_C \,\|\, \mathsf{GS} \\
C'
\end{array}.
$$

We apply the induction hypothesis to $K_Y \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, P(\!|M_1[A]_{S'}, \varnothing, M_3,\ldots, M_n|\!)$ and get K and $C''[\cdot]_{R''}$, such that

$$
\begin{array}{c}
\varnothing \\
\mathcal{D}''_C \,\|\, \mathsf{GS} \\
C''
\end{array}
,\quad
\begin{array}{c}
\varnothing \\
\mathcal{D}_A \,\|\, \mathsf{GS} \\
K \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, A
\end{array}
\quad\text{and}\quad
\begin{array}{c}
C''[K \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, \mathcal{X}]_{R''} \\
\mathcal{D}''_G \,\|\, \mathsf{GS} \\
K_Y \,\mathbin{\rotatebox[origin=c]{180}{$\&$}}\, P(\!|M_1[\mathcal{X}]_{S'}, \varnothing, M_3,\ldots, M_n|\!)
\end{array}
$$

for any $\mathcal{X}$. We let $C[\cdot]_R = C'[C''[\cdot]_{R''}]_{R'}$, the derivation $\mathcal{D}_C$ be defined by Lemma A.1 and the derivation $\mathcal{D}_G$ defined as follows

$$
\begin{array}{c}
C'\left[
\begin{array}{c}
\underset{\text{s.sw}\downarrow}{\underline{K_Y \bindnasrepma P\left(\left[M_1[\mathcal{X}]_{S'}, \begin{array}{c} \varnothing \\ \mathcal{D}_X \,\| \\ K_X \bindnasrepma M_2 \end{array}, M_3, \dots, M_n\right)\right)}} \\
\begin{array}{c} C''[K \bindnasrepma X]_{R''} \\ \mathcal{D}'_G \,\| \\ \end{array}
\end{array}
\right]_{R'}
\end{array}
$$

$$
\text{s.sw}\downarrow \frac{\qquad\qquad\qquad}{\begin{array}{c} C'[K_X \bindnasrepma K_Y]_{R'} \\ \mathcal{D}''_G \,\| \\ G'' \end{array} \quad \bindnasrepma P\left(M_1[\mathcal{X}]_{S'}, M_2, M_3, \dots, M_n\right)}
$$

$\square$

Finally, it remains to give the proof of Atomic Splitting, which follows from context reduction (Lemma 6.6) and splitting for prime graphs (Lemma 6.3), as indicated in Figure 3.

Lemma 6.5 (Atomic Splitting). *Let G be a graph. If $\vdash_{\mathsf{GS}} G \bindnasrepma a$ for an atom a, then there is a context $C[\cdot]_R$ such that there are derivations*

$$
\begin{array}{c} C[a^\perp]_R \\ \mathcal{D}_G \,\|\, \mathsf{GS} \\ G \end{array} \quad and \quad \begin{array}{c} \varnothing \\ \mathcal{D}_C \,\|\, \mathsf{GS} \\ C \end{array} \quad .
$$

Proof. The proof is similar to the one of Lemma 6.3. We assume $\vdash_{\mathsf{GS}} G \bindnasrepma a$ and aim to construct $C[\cdot]_R, \mathcal{D}_G, \mathcal{D}_C$ as in the statement of the lemma. Observe that $G \neq \varnothing$, otherwise $G \bindnasrepma a$ would not be provable in GS. We make a case analysis on the bottommost rule instance r in $\mathcal{D}$, and follow the same pattern as in the case analysis in the proof of Lemma 6.3.

(a) If rule r acts inside G, then the derivation $\mathcal{D}$ is of shape

$$
\begin{array}{c} \varnothing \\ \mathcal{D}' \,\|\, \mathsf{GS} \\ \mathsf{r}\dfrac{G'}{G} \bindnasrepma a \end{array}
$$

for some $\mathcal{D}'$. By Observation 5.14 we know that $\|G'\| < \|G\|$, then we apply the induction hypothesis on $G' \bindnasrepma a$. This gives us a context $C'[\cdot]_{R'}$ and two derivations

$$
\begin{array}{c} C'[a^\perp]_R \\ \mathcal{D}'_G \,\|\, \mathsf{GS} \\ G' \end{array} \quad and \quad \begin{array}{c} \varnothing \\ \mathcal{D}'_C \,\|\, \mathsf{GS} \\ C' \end{array} \quad .
$$

We conclude by applying the rule r to the conclusion G' of $\mathcal{D}_{G'}$.

(b) The last rule in $\mathcal{D}$ is a s.sw↓, such that $\mathcal{D}$ is of the following shape (where $G = G'' \parr G'$ and $G' \neq \varnothing$):

$$\text{s.sw↓} \; \frac{\begin{array}{c} \varnothing \\ \mathcal{D}' \parallel \mathsf{GS} \\ G'' \parr a[G']_S \end{array}}{G'' \parr G' \parr a}$$

If the s.sw↓ instance is not trivial, then $G'' \parr a[G']_S \simeq G'' \parr (a \otimes G') \simeq G[a]_{V_{G'}}$, which makes this a case of Case (c) below.

(c) The last rule in $\mathcal{D}$ is a s.sw↓, such that $\mathcal{D}$ is of the shape

$$\text{s.sw↓} \; \frac{\begin{array}{c} \varnothing \\ \mathcal{D}' \parallel \mathsf{GS} \\ G[a]_S \end{array}}{G \parr a}$$

By Lemma 6.6 there is a graph K and a context $C[\cdot]_R$ such that there are derivations

$$\begin{array}{c} \varnothing \\ \mathcal{D}'_C \parallel \mathsf{GS} \\ C' \end{array} \quad \text{and} \quad \begin{array}{c} \varnothing \\ \mathcal{D}^* \parallel \mathsf{GS} \\ K \parr a \end{array} \quad \text{and} \quad \begin{array}{c} C'[K \parr \mathcal{X}]_R \\ \mathcal{D}_G^{\mathcal{X}} \parallel \mathsf{GS} \\ G[\mathcal{X}]_S \end{array}$$

for any graph $\mathcal{X}$. Now we apply the induction hypothesis to $\mathcal{D}^*$ to obtain a context $C'''[\cdot]_{R''}$ such that we have

$$\begin{array}{c} C'''[a^\perp]_{R''} \\ \mathcal{D}_K \parallel \mathsf{GS} \\ K \end{array} \quad \text{and} \quad \begin{array}{c} \varnothing \\ \mathcal{D}''_C \parallel \mathsf{GS} \\ C'' \end{array}$$

We let $C[\cdot]_R = C'[C''[\cdot]_{R''}]_{R'}$ and construct $\mathcal{D}_G$ as

$$\begin{array}{c} C' \left[\begin{array}{c} C'''[a^\perp]_{R''} \\ \mathcal{D}_K \parallel \\ K \end{array} \right]_{R'} \\ \mathcal{D}_G^{\varnothing} \parallel \\ G[\varnothing]_S \end{array}$$

(Note that $K \simeq K \parr \varnothing$ and $G \simeq G[\varnothing]_S$.)

(d) The last rule in $\mathcal{D}$ is an ai↓ such that $\mathcal{D}$ is of shape

$$\begin{array}{c} \varnothing \\ \mathcal{D}'_G \parallel \mathsf{GS} \\ G' \parr \; \text{ai↓} \, \dfrac{\varnothing}{a^\perp \parr a} \end{array}$$

i.e., $G \simeq G' \parr a^\perp$. We can conclude by letting $C = \varnothing$ (hence $\mathcal{D}_C$ is trivial) and $\mathcal{D}_G = \mathcal{D}'_G \parr a^\perp$.

(e) If the last rule is a $\mathsf{p}{\downarrow}$ which does not act inside G, such that $G \simeq G' \,⅋\, P(\!|N_1, \ldots, N_m|\!)$, then w.l.o.g. $\mathcal{D}$ is of the shape

$$
G' \,⅋\, \mathsf{p}{\downarrow}\frac{\overset{\varnothing}{\underset{\mathcal{D}'}{\,\|\,}\mathsf{GS}}\quad ((N_1 \,⅋\, a) \otimes N_2 \otimes \cdots \otimes N_m)}{P(\!|N_1, \ldots, N_m|\!) \,⅋\, a} \quad .
$$

By Lemma 6.7 there is a context $C'[\cdot]_{R'}$ and graphs $L_1, \ldots, L_m$ such that there are derivations

$$
\begin{array}{cccc}
\underset{\underset{G'}{\mathcal{D}'_G\,\|\,\mathsf{GS}}}{C'[L_1 \,⅋\, \cdots \,⅋\, L_m]_{R'}} & \underset{\underset{L_1 \,⅋\, (a \,⅋\, N_1)}{\mathcal{D}_1\,\|\,\mathsf{GS}}}{\varnothing} & \underset{\underset{L_i \,⅋\, N_i}{\mathcal{D}_i\,\|\,\mathsf{GS}}}{\varnothing} \text{ and } & \underset{\underset{C'}{\mathcal{D}'_C\,\|\,\mathsf{GS}}}{\varnothing} \quad .
\end{array}
$$

for all $i \in \{2, \ldots, m\}$. By inductive hypothesis on $L_1 \,⅋\, (a \,⅋\, N_1)$ we have

$$
\underset{\underset{L_1 \,⅋\, N_1}{\mathcal{D}'_1\,\|\,\mathsf{GS}}}{C''[a^\perp]_{R''}} \quad \text{and} \quad \underset{\underset{C''}{\mathcal{D}''_C\,\|\,\mathsf{GS}}}{\varnothing} \quad .
$$

for some context $C''[\cdot]_{R''}$. We conclude by setting $C = C'[C''[\cdot]_{R''}]_{R'}$ since

$$
C'\left[\mathsf{s.sw}{\downarrow}\frac{\left(\mathsf{s.sw}{\downarrow}\dfrac{\underset{\underset{L_1 \,⅋\, N_1}{\mathcal{D}'_1\,\|}}{C''[a^\perp]_{R''}} \otimes \underset{\underset{L_2 \,⅋\, N_2}{\mathcal{D}'_2\,\|}}{\varnothing} \otimes \cdots \otimes \underset{\underset{L_m \,⅋\, N_m}{\mathcal{D}'_m\,\|}}{\varnothing}}{L_1 \,⅋\, \cdots \,⅋\, L_m \,⅋\, \mathsf{p}{\downarrow}\dfrac{N_1 \otimes N_2 \otimes \cdots \otimes N_m}{P(\!|N_1, \ldots, N_m|\!)}}\right)}{\underset{\underset{G'}{\mathcal{D}'_G\,\|}}{C'[L_1 \,⅋\, \cdots \,⅋\, L_m]_{R'}} \,⅋\, P(\!|N_1, \ldots, N_m|\!)}\right]_{R'}
$$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $\square$

Appendix B. Requirements of an Analytic Proof System on Graphs

In this section, we reflect on design decisions in order to support our claim that we have defined a logic where we reason about graphs rather than formulas. We present here an argument that is independent from the proof system that we developed, thus it is not necessary for the reader to accept deep inference a priori in order to accept the set of theorems proven by the logic GS. We reinforce the message that, for this logic without precedent (that is without a pre-existing semantics or proof system), we had success designing and justifying our system when we started from logical principles. Our approach of designing from principles is credible, since, as highlighted in the discussion on related work in Section 10,

there does not appear to be an immediate generalisation of the semantics for some established logic on formulas to graph that yields a system with the logical properties we desire.

From the title, we desire:

An analytic propositional proof system on graphs.

To understand fully the above statement we explain its two aspects. Firstly, we make precise what we mean by a *propositional proof system*, by means of logical principles that such a proof system should conform to. Secondly, we must explain what it means for such a propositional proof system to be *analytic*, particularly since traditional definitions of analyticity do not lift immediately to our setting. In the discussion that follows we show that all design decisions are widely accepted in logic, making is difficult to argue that GS is not a logic.

Graph isomorphism for propositional proofs. In this study, we have restricted ourselves to simple undirected graphs (see Definition 2.1), where vertices are labelled with positive or negative propositional atoms, such as a and $a^\perp$. This allows us to align with existing graphical representations of formulas if we restrict to *cographs*, which are exactly those graphs generated by propositional formulas using the operations illustrated in (2.3).

The first logical assumption we make is that isomorphic graphs (see Definition 2.2) are logically equivalent, which we expect to hold for all graphical logics.

> **Extensionality requirement:** For pairs of isomorphic graphs G and H, we have that $\vdash G \multimap H$ holds.

Graph isomorphisms allows us to rename the underlying vertices of a graph, while preserving labels. Indeed in our diagrams of graphs, extensionality is implicitly appealed to, since we only show the labels of vertices, i.e., we quotient graphs by label-preserving isomorphisms. The term "extensionality" is consistent with the idea that objects are equivalent if their externally visible properties, i.e., their labels and edges shown in diagrams, are the same.

This is one of the few principles we expect to be common to all logics on graphs. To reinforce this belief, we acknowledge there are schools of philosophy that maintain that it is possible that $A \neq A$ [Mat68]. The essence of such arguments is that if A is not well typed or does not exist then $A \neq A$ is a reasonable conclusion. But notice that, firstly, this is different from saying that $A = A$ does not hold, and, secondly, a metaphysical discussion on types or existence is perpendicular to the logic in this work.

Even if we assume extensionality as our sole logical principle, since we aim to define a propositional proof system for our logic on graphs, we must take additional care to ensure that we satisfy the following principle.

> **Cook-Reckhow requirement:** Every rule is checkable in polynomial time.

The above is a fundamental property of proof systems for propositional logic [CR79]. As mentioned in Observation 5.12, since checking an explicit isomorphism is in P but currently there are no algorithms for finding graph isomorphisms in P, a formulation of a propositional proof system on graphs must make the isomorphisms explicit in the proof system.

Involutive negation and consistency. The first proper design decision we make is that we insist on having a logic featuring an involutive negation, as found in most classical and linear logics (but not intuitionistic logic of course).

> **De Morgan requirement:** Negation should be involutive.

Formally, an involution on graphs is a unary operator $(\cdot)^\perp$ that satisfies the property $(G^\perp)^\perp = G$ for all graphs G. The assumption that we have an involutive negation means that De Morgan dualities hold for pairs of connectives on graphs we define (Observation 4.3). For a proof system on graphs, there are only two possible choices for an involution, namely the identity function and the graph complement function.

The use of the identity function to define an involutive negation is ruled out by the logical principle of consistency, which is that not all graphs are provable. The formulation of consistency that we achieve for GS is as follows.

> **Consistency requirement:** For non-empty graphs G, if $\vdash G$ then $\nvdash G^\perp$.

To see why, the above principle rules out the identity function as negation, observe that extensionality ensures that there are some provable graphs say $\vdash G$. If the negation is defined by the identity, then $\vdash G^\perp$ holds, which violates the above consistency requirement. Thereby, the assumptions thus far fix negation as *graph complement* (Definition 4.1). Observe also that the consistency of GS is indeed a corollary of cut elimination (Corollary 5.11).

Implication. Our next proper design decision is to materialise implication in our logic. We materialise implication "G implies H" as "not G or H", for some notion of disjunction, as in logics such as classical and linear logics. We know already that negation is graph complement, but disjunction we have not yet defined. In order for there to be a single implication, the disjunction used must be commutative (implications materialised using non-commutative disjunction lead to a distinct left and right implication [Lam61]).

> **AC requirement:** Disjunction is commutative and associative.

While there may be several elaborate choices for defining disjunction,[15] we make the design decision that disjunction is defined as the disjoint union of graphs. This design decision aligns with an established culture of using cographs to represent formulas [Duf65, Ret03]. Note that, by symmetry, we could have selected graph join as disjunction, which would simply interchange edges and non-edges throughout this paper without changing the meaning of the logic.

We are now able to materialise implication by means of negation and disjunction, as in classical and linear logic. More precisely, we have the following principle.
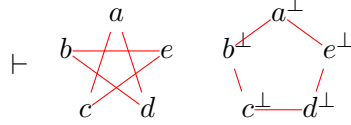
> **Material requirement:** Implication $G \multimap H$ is defined as $G^\perp \,\invamp\, H$.

This assumption should not be taken for granted, since various logics such as intuitionistic logic cannot materialise implication in this way. Also, this is not a guaranteed property of a logic satisfying De Morgan properties, since even if a disjunction is present it may not be the right disjunction to internalise negation. For example, linear implication cannot be materialised using additive disjunction and linear negation, hence the additive fragment of linear logic [Gir87a] has no material implication.

We can now, by using only the principals we have laid out above, prove important examples independently of the proof system developed in the body of this paper. By extensionality we have implication $\vdash a \multimap a$, where a is the singleton graph. Hence we know that $\vdash a^\perp \,\invamp\, a$ should hold in our logic. This of course, agrees with the majority of proof systems. Moving beyond graphs that correspond to traditional formulas, we also have

---

[15]For an example, consider the way multiplicative disjunction is defined on coherent spaces [GLT89].

theorems such as the following, by the same reasoning, i.e., the two disjoint sub-graphs are dual.

For almost all logical systems, we require that implication is transitive, since transitivity, known since antiquity as the *hypothetical syllogism*, enables us to perform deduction.

Transitivity requirement: If $\vdash F \multimap G$ and $\vdash G \multimap H$, then $\vdash F \multimap H$.

Notice that transitivity allows us to apply *modus ponens*. To be explicit, observe that the empty graph (denoted $\varnothing$) is the unit of the disjoint union operation on graphs. Furthermore $\varnothing$ is self-dual. By the definition of implication, we have that $\vdash G$ holds whenever $\vdash \varnothing \multimap G$ does. Thereby, if we assume $\vdash G$ and $\vdash G \multimap H$ hold, then by the definition of implication $\vdash \varnothing \multimap G$ holds and by transitivity $\vdash \varnothing \multimap H$ holds. By the same argument we can conclude that $\vdash H$ holds. This is exactly modus ponens.

Like consistency, transitivity of implication is a default decision, although we acknowledge that transitivity can fail for some logics featuring negation-as-failure [BM90].

The final feature we add is the context-free assumption. For a logic with formulas this is the assumption that implication is preserved in all contexts, where a context is a formula with a hole in which another formula can be plugged, such as $\phi \otimes [\cdot]$. For example, if $\psi \multimap \chi$, then $\phi \otimes \psi \multimap \phi \otimes \chi$. More specifically, we mean all positive contexts, since negations reverse the direction of implication. However, in a logic satisfying the assumptions we have made, with De Morgan dualities and material implication, we can always reduce to a negation-normal-form, where all negations are pushed to the atoms, i.e., the labels on the vertices; thereby allowing us to range over all contexts. Similarly, in modal logics extending K we have that if $\psi \multimap \chi$ then $\Box\psi \multimap \Box\chi$. The same holds for the diamond modality. Indeed, this holds even for non-classical modal logics [MS14].

When we move from formulas to graphs the notion of a context must be generalised, where the obvious notion is introduced in Notation 3.3. In the broader philosophy of language (programming languages and natural languages), context freedom is not guaranteed, since the meaning of something may change in different contexts. However, logics are usually designed such that if there is some context in which an implication does not hold, then the implication itself does not hold; situations in knowledge representation where this fails is usually due to using moving between systems [Gab98].

Context-free requirement: For all positive contexts $C[\cdot]_R$, if $\vdash G \multimap H$, then we have $\vdash C[G]_R \multimap C[H]_R$.

The above property ensures that if we prove a theorem then it holds in any context. It is exploited explicitly in some proof systems to allow rules to be applied deep inside any module of a graph, which is the technique called *deep inference* employed in this paper. However, the above principle is not specific to deep inference.

Thus we expect the following to be a theorem of a logic satisfying the above principles.

$$\vdash (G \multimap H) \multimap (C[G] \multimap C[H]) \tag{B.1}$$

From this, we can establish non-trivial facts that we expect to hold for a logic on graphs, that are beyond the scope of formulas. Consider the following example where we instantiate (B.1)

with

$$G \triangleq a \qquad H \triangleq \varnothing \qquad C[\cdot]_{\{a\}} \triangleq$$

From the above instantiation we obtain the following theorem:

$$\vdash (c \multimap \varnothing) \multimap \left(\quad \multimap \quad \right)$$

Reorganising, according to the definitions of implication and negation that we have fixed, we obtain one of our running examples (see (4.9), (4.13) and (6.3)).

$$\vdash \qquad \multimap$$

The above proposition would hold for any logic satisfying the principles laid out in this section. There may be more propositions that hold other than those that are enforced by these principles; for instance, if we induce more principles from classical logic, we could prove strictly more theorems. However, in this work, we make the design decision of aiming for the minimal system, which we state as a concluding principle that fixes our target logic.

> **Minimality requirement:** No further propositions hold, other than those forced by the other requirements.

We have remarked throughout this section that GS satisfies all the principles laid out, mainly as a consequence of cut elimination. Minimality then follows from observing that all rules of GS are sound with respect to the same principles.

On analyticity. Throughout this paper we provide evidence that we can design *propositional proof systems* that achieve the above stated requirements. However, designing a proof system is not the main challenge. The main challenge is to design an *analytic* propositional proof system.

Analyticity, the idea that propositions contains all the information required to in order to judge their validity with respect to some logical system, has long been debated by philosophers. The design of modern analytic proof calculi is widely considered to begin with the *sequent calculus*, as developed in 1935 [Gen35a, Gen35b], with improvements incorporated by Girard [Gir87a].

The rules of the sequent calculus, such as those for MLL° presented in Figure 6, are considered to be analytic since they satisfy the *subformula property*. The subformula property states that in each rule, every formula that occurs in one premise sequent, occurs as a subformula in the conclusion sequent. The subformula property facilitates proof search, since there are only finitely many subformulas to consider during proof search. The rules in Figure 6 clearly satisfy the subformula property; whereas the cut rule, below, which generalises transitivity, and hence also *modus ponens*, does not in general satisfy the subformula property.

$$\mathsf{cut} \frac{\Gamma, \phi \qquad \phi^\perp, \Delta}{\Gamma, \Delta}$$

If we start with some conclusion Γ, Δ and try to apply the cut rule above, there are infinitely many formulas ϕ to choose from. Similarly, the transitivity assumption requires insight

external to the system to know which formulas to introduce. Thus the subformula property is effectively avoiding infinite branching in the proof search space. This is a fundamental reason for proving cut elimination when designing a proof system: on one hand, we aim to ensure that the basic principles of deduction such as modus ponens may be applied; on the other hand, we also want to show that deductive rules such as cut and modus ponens, which are not well-behaved with respect to proof search are *admissible*.

The sub-formula property does not lift immediately to the setting of deep inference. Since deep inference is necessary for GS, we make use of an alternative definition of analyticity [BG09, BG16].

Analyticity requirement: For graph G and rule r, there is an n such that,

for all contexts $C[\cdot]_R$, there are at most n graphs H such that $\mathsf{r}\dfrac{C[H]_R}{C[G]_R}$.

This ensures a rule is guaranteed to be finitely branching regardless of the context in which the rule is applied. This property follows immediately for GS from inspecting the rules.

Thus, in this work, we have shown that we can design a proof system on graphs satisfying a notion of analyticity. The fact that traditional methods (such as the sequent calculus) for developing analytic proof calculi fail here is a surprise, particularly because we have targeted the minimal logic on graphs satisfying some widely accepted logical principles. The logical principles we have based our design decisions on we consider to be difficult to argue against — which does not prevent one from exploring alternative design decisions that may lead to further logics on graphs. The only assumption that may be strongly argued about is the use of the empty graph as a self-dual unit; which can be regarded as a simplifying assumption in our initial design. That design decision can be justified by *conservativity* (Theorem 8.6), in the sense that there are established formula-based logics featuring such a self-dual unit.