



HAL
open science

A System of Interaction and Structure III: The Complexity of BV and Pomset Logic

Lê Thành Dũng Nguyễn, Lutz Straßburger

► **To cite this version:**

Lê Thành Dũng Nguyễn, Lutz Straßburger. A System of Interaction and Structure III: The Complexity of BV and Pomset Logic. Logical Methods in Computer Science, 2023, Volume 19, Issue 4, 10.46298/LMCS-19(4:25)2023 . hal-03909547v1

HAL Id: hal-03909547

<https://inria.hal.science/hal-03909547v1>

Submitted on 21 Dec 2022 (v1), last revised 10 Jan 2024 (v2)

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

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



Distributed under a Creative Commons Attribution 4.0 International License

A SYSTEM OF INTERACTION AND STRUCTURE III: THE COMPLEXITY OF BV AND POMSET LOGIC*

LÊ THÀNH DŨNG (TITO) NGUYỄN ^a AND LUTZ STRASSBURGER ^b

^aLaboratoire de l'informatique du parallélisme (LIP), École Normale Supérieure de Lyon, France
e-mail address: nlttd@nguyentito.eu

^bInria Saclay & École Polytechnique, Palaiseau, France

ABSTRACT. Pomset logic and BV are both logics that extend multiplicative linear logic (with Mix) with a third connective that is self-dual and non-commutative. Whereas pomset logic originates from the study of coherence spaces and proof nets, BV originates from the study of series-parallel orders, cographs, and proof systems. Both logics enjoy a cut-admissibility result, but for neither logic can this be done in the sequent calculus. Provability in pomset logic can be checked via a proof net correctness criterion and in BV via a deep inference proof system. It has long been conjectured that these two logics are the same.

In this paper we show that this conjecture is false. We also investigate the complexity of the two logics, exhibiting a huge gap between the two. Whereas provability in BV is NP-complete, provability in Pomset logic is Σ_2^P -complete. We also make some observations with respect to possible sequent systems for the two logics.

CONTENTS

1. Introduction	2
2. Preliminaries on Pomset Logic and BV	4
2.1. Formulas, Duality and Sequents	4
2.2. Dicographs, Relation Webs and Series-Parallel Orders	6
2.3. Perfect Matchings, RB-digraphs and Alternating Elementary Cycles	10
2.4. Pomset Logic, Proof Nets and Balanced Formulas	11
2.5. System BV and the Calculus of Structures	14
2.6. Unit-Free Versions of BV and SBV	16
2.7. SBVu and Dicograph Inclusions	19

Key words and phrases: proof nets, deep inference, pomset logic, system BV, cographs, dicographs, series-parallel orders, relation webs.

* This article extends our previous CSL 2022 paper [NS22] with a substantial amount of new material. It has been invited to a special issue of LMCS.

L. T. D. Nguyễn was supported by the LambdaComb project (ANR-21-CE48-0017) while working at École Polytechnique and by the LABEX MILYON (ANR-10-LABX-0070) of Université de Lyon, within the program “Investissements d’Avenir” operated by the French National Research Agency (ANR).

3.	Comparing BV and Pomset Logic	20
3.1.	BV is Contained in Pomset Logic	20
3.2.	Pomset Logic is not Contained in BV	23
4.	Complexity of Provability	27
4.1.	Preliminaries on Complexity Theory and Boolean Formulas	28
4.2.	BV is NP-Complete (and how to Generalize Membership in NP)	30
4.3.	Correctness of Pomset Logic Proof Nets is coNP-Complete	32
4.4.	Finding Alternating Elementary Cycles is NP-Hard	35
4.5.	Pomset Logic Provability is Σ_2^P -Complete	40
5.	Back to the Sequent Calculus	44
5.1.	Retoré’s Sequent Calculus with Cuts is Equivalent to BV	45
5.2.	A Refinement of Tiu’s Argument	48
5.3.	A Complexity-Theoretic Obstacle to Sequent Calculi for Pomset Logic	49
5.4.	A Reconstruction of Slavnov’s Calculus	51
6.	Conclusion	53
6.1.	Related topics	54
6.2.	Open problems	56
	References	57

1. INTRODUCTION

There are two ways to put this paper into perspective. First, it is the third paper in a series of five papers: the first two [Gug07, Tiu06] introduce a logic called *system BV*, prove cut elimination for it, and show that *deep inference* is necessary for having a cut-free deductive system for BV; and the last two [SG11, GS11] study system NEL, a conservative extension of BV with the exponentials of linear logic. The second way to look at this paper is that it finally answers the longstanding open question whether BV and pomset logic [Ret21] are the same.

Pomset logic has been discovered by Christian Retoré [Ret93, Ret97a] through the study of coherence spaces which form a semantics of proofs for linear logic [Gir87], by observing¹ that next to the two operations \otimes (*tensor* or *multiplicative conjunction*) and \wp (*par* or *multiplicative disjunction*) on coherence spaces there are two other operations \triangleleft and \triangleright , which are non-commutative, obey $A \triangleleft B = B \triangleright A$, and are self-dual, i.e., $\langle A \triangleleft B \rangle^\perp = A^\perp \triangleleft B^\perp$.² Furthermore, there are linear maps $A \otimes B \multimap A \triangleleft B$ and $A \triangleleft B \multimap A \wp B$. From this semantic observation, Retoré derived a *proof net* syntax, i.e., a graph-like representation of proofs together with a combinatorial *correctness criterion* that distinguishes actual proofs among the considered space of graphs, together with a cut elimination theorem. However, he could not provide a sound and complete cut-free sequent calculus for this logic.

¹According to Retoré (cf. <http://www.lirmm.fr/~retore/PRESENTATIONS/pomset4michele.pdf>), this semantic observation is due to Girard.

²Observe that the order is *not* inverted, as it is the case with other non-commutative logics, cf. §6.1.

System BV was found by Alessio Guglielmi [Gug07, Gug99]³ through a syntactic investigation of the connectives of pomset logic and a graph theoretic study of series-parallel orders and cographs. The difficulty of presenting this combination of commutative and non-commutative connectives in the sequent calculus triggered the development of the *calculus of structures* [GS01], the first proper deep inference proof formalism.⁴

This leads to the strange situation that we have two logics, pomset logic and BV, which are both conservative extensions of multiplicative linear logic with mix (MLL+mix) [FR94] with a noncommutative connective \triangleleft such that $A \otimes B \multimap A \triangleleft B \multimap A \wp B$, and which both obey a cut elimination result—the only difference being that pomset logic only had a proof net syntax but no deductive proof system, and BV only had a deductive proof system but no proof nets. This naturally led to the conjecture that both logics are the same [Str03b].

In this paper we show that this conjecture is false. It can easily be shown [Str03b, Str03a, Ret99b] that every theorem in BV is also a theorem of pomset logic, and for the sake of clarity, we also give a proof in this paper (in Section 3.1). However, the converse is not true, and we give an example of a formula that is a theorem of pomset logic but not provable in BV (in Section 3.2).

This naturally leads to the question of complexity. Do both logics have the same or different complexity? It has been observed in [Kah08] that provability in BV is NP-complete. The reason is that NP-hardness is inherited from MLL+mix, and containment in NP follows from the fact that the size of every proof in BV is polynomial in the size of the conclusion, and the correctness of such a proof can be checked in time which is linear in the size of the proof (see Section 4.2 for more details). However, provability in pomset logic is Σ_2^P -complete. Even though the size of a pomset logic proof net is polynomial in the size of its conclusion, checking correctness of such a proof net is coNP-complete. The details for these results are discussed in Sections 4.3–4.5.

This complexity result explains why it is impossible to give a deductive proof system (in the sense of Cook and Reckhow [CR79]) for pomset logic. Nonetheless, there is a recent proposal by Slavnov [Sla19] for a decorated sequent calculus for pomset logic. We look at this calculus in Section 5.4 and relate it to our complexity result. Before that, we show in Section 5.1 that the sequent calculus with cut, that Retoré proposed for pomset logic is in fact a sound and complete sequent calculus for BV.⁵

In summary, the paper makes the following contributions:

- In Section 2, we give a gentle and easy accessible introduction to pomset logic and BV. We unify the notation and terminology, and present some important properties, that will be needed in later sections.
- In Section 3, we are showing that BV is properly contained in pomset logic, by showing that every theorem of BV is a theorem of pomset logic, but not vice versa. The results of this section have already been presented in [NS22], of which this article is an extended version.
- In Section 4, we are discussing the complexity of pomset logic and BV. More precisely, we are recalling Kahramanoğulları’s result on the NP-completeness of BV, and we show

³The initial ideas of the inference rules have also been present in [Ret99b] even though not formulated as a proof admitting cut-elimination.

⁴An introductory survey on deep inference can be found in [TS19].

⁵In fact, in various publications on pomset logic, Retoré proposes different sequent calculi (one in his PhD thesis [Ret93, Chapitre 8], one in [Ret97a, Section 7], and more recently another one in [Ret21]); we work here with the one in [Ret21].

$$\begin{array}{lll}
A \otimes (B \otimes C) \equiv (A \otimes B) \otimes C & A \otimes B \equiv B \otimes A & \mathbb{I} \otimes A \equiv A \\
A \wp [B \wp C] \equiv [A \wp B] \wp C & A \wp B \equiv B \wp A & \mathbb{I} \wp A \equiv A \\
A \triangleleft \langle B \triangleleft C \rangle \equiv \langle A \triangleleft B \rangle \triangleleft C & & \mathbb{I} \triangleleft A \equiv A \equiv A \triangleleft \mathbb{I}
\end{array}$$

FIGURE 1. The equations defining \equiv .

that checking correctness of a pomset logic proof is **coNP**-complete and that provability in pomset logic is Σ_2^P -complete.

- Finally, in Section 5, we come back to the sequent calculus, discussing the difficulties that the two logics pose, and how they can or cannot be overcome.

2. PRELIMINARIES ON POMSET LOGIC AND BV

In this section we will introduce the the two logics, **BV** and pomset logic, together with some basic underlying graph theoretical and proof theoretical concepts. Even though pomset logic was discovered through the study of coherence semantics, we will here only discuss its syntax, as coherence spaces are not needed for the results of this paper.

2.1. Formulas, Duality and Sequents. The *formulas* of pomset logic and **BV** are in this paper denoted by capital Latin letters A, B, C, \dots and are generated from propositional variables a, b, c, \dots , their *duals* $a^\perp, b^\perp, c^\perp \dots$ and the *unit* \mathbb{I} via the three binary connectives *tensor* \otimes , *par* \wp , and *seq* \triangleleft . We shall also be led to consider terms built with those connectives whose leaves are taken in arbitrary sets, which justifies the following more general definition.

Definition 2.1. The *generalized formulas* over the set X are generated by the grammar

$$A, B ::= \mathbb{I} \mid x \mid (A \otimes B) \mid [A \wp B] \mid \langle A \triangleleft B \rangle \quad \text{where } x \in X$$

We fix a countable set $\mathcal{V} = \{a, b, c, \dots\}$ of *propositional variables*. To each variable a we injectively associate a *dual* a^\perp ; we write $\mathcal{V}^\perp = \{a^\perp \mid a \in \mathcal{V}\}$, and require that $\mathcal{V} \cap \mathcal{V}^\perp = \emptyset$. An *atom* is either a variable (*positive atom*) or the dual of a variable (*negative atom*). A *formula* is a generalized formula over the set of atoms $\mathcal{V} \cup \mathcal{V}^\perp$.

A generalized formula over X is *linear* when it contains at most one occurrence of each $x \in X$. In particular, when X is the set of atoms, a linear formula contains at most one positive occurrence a and at most one negative occurrence a^\perp of each propositional variable $a \in \mathcal{V}$.

Definition 2.2. The *size* of a generalized formula A over the set X , denoted by $|A|$, is the number of occurrences of elements of X in A .

For better readability of large formulas, we use here different kinds of parentheses for the different connectives.⁶ In the following, we omit outermost parentheses for better readability.

Definition 2.3. We define the *relation \equiv on generalized formulas* to be the smallest congruence generated by the rules of Figure 1. Those correspond to associativity of $\otimes, \wp, \triangleleft$, commutativity of \otimes, \wp , and unit equations (\mathbb{I} behaves as unit for all three connectives).

As usual in the linear logic tradition, negation is defined not as a connective but as a mapping from formulas to formulas. Note that this does not make sense for generalized formulas.

Definition 2.4. The involutive (*linear*) *negation* or *duality* $(-)^{\perp}$ is extended from propositional variables to formulas by taking De Morgan’s laws as its inductive definition:

$$(a^{\perp})^{\perp} = a \quad \mathbb{I}^{\perp} = \mathbb{I} \quad (A \otimes B)^{\perp} = A^{\perp} \wp B^{\perp} \quad [A \wp B]^{\perp} = A^{\perp} \otimes B^{\perp} \quad \langle A \triangleleft B \rangle^{\perp} = A^{\perp} \triangleleft B^{\perp}$$

The first clause means that linear negation defined a fixed-point-free involution on the set of atoms. The last clause is what we mean when we say that seq is *self-dual*; note that the right-hand side is indeed $A^{\perp} \triangleleft B^{\perp}$ and not $B^{\perp} \triangleleft A^{\perp}$.⁷

We will also need the notion of *sequent* in pomset logic. While traditional sequent calculi use multisets⁸ of formulas as sequents, pomset logic is thus named because its sequents are *partially ordered multisets* of formulas. While Retoré’s early work [Ret93, Ret97a] involved arbitrary partial orders, we consider here the simplified version from [Ret21] where sequents are equipped with *series-parallel* orders. We shall define those orders in the next subsection (see Proposition 2.31 and Remark 2.32); for now, let us just say that those orders admit a syntactic description that we give here.

Definition 2.5. We denote a *sequent* in pomset logic by capital Greek letters Γ, Δ, \dots and they are generated as follows:

$$\Gamma, \Delta ::= \emptyset \mid A \mid [\Gamma, \Delta] \mid \langle \Gamma; \Delta \rangle$$

where \emptyset stands for the empty sequent and A can be any formula. As before, we use different kinds of brackets for better readability. Furthermore, we consider sequents equal modulo commutativity of $[\cdot, \cdot]$ and associativity of $[\cdot, \cdot]$ and $\langle \cdot; \cdot \rangle$, and the unit laws for the empty sequent:

$$\begin{aligned} [\Gamma, \Delta] &= [\Delta, \Gamma] & [\Gamma, \emptyset] &= \Gamma & \langle \Gamma; \emptyset \rangle &= \Gamma = \langle \emptyset; \Gamma \rangle \\ [\Gamma, [\Delta, \Lambda]] &= [\Gamma, \Delta, \Lambda] = [[\Gamma, \Delta], \Lambda] & \langle \Gamma; \langle \Delta; \Lambda \rangle \rangle &= \langle \Gamma; \Delta; \Lambda \rangle = \langle \langle \Gamma; \Delta \rangle; \Lambda \rangle \end{aligned}$$

In the remainder of this paper we will always omit redundant brackets.

Definition 2.6. A sequent is called *flat* iff it does not contain the $\langle \cdot; \cdot \rangle$ constructor. Therefore, flat sequents can be described by multisets of formulas.

Remark 2.7. Pomset logic is not the only system that features “non-flat” sequents with two distinct connectives. Another famous example is the logic **BI** of bunched implications [OP99].

The operations $[\cdot, \cdot]$ and $\langle \cdot; \cdot \rangle$ serve as counterparts on sequents to the connectives \wp and \triangleleft on formulas (just as the sequent $\vdash A, B, C$ morally means $A \vee B \vee C$ in classical logic). More formally:

Proposition 2.8. *A sequent is the same thing as a generalized \otimes -free formula over the set of formulas, modulo \equiv , writing $\emptyset, [\Gamma, \Delta]$ and $\langle \Gamma; \Delta \rangle$ instead of $\mathbb{I}, \Gamma \wp \Delta$ and $\Gamma \triangleleft \Delta$ respectively.*

⁶Note that this is redundant and carries no additional meaning. The only use is better readability.

⁷In that respect, pomset logic and BV are different from other non-commutative variants of linear logic where \otimes and \wp are non-commutative with $(A \otimes B)^{\perp} = B^{\perp} \wp A^{\perp}$, see Section 6.1.

⁸This holds for commutative logics. Indeed, using multisets is equivalent to using lists and adding the exchange rule to the system.

It is then natural to define a collapse operation⁹ from sequents into formulas. But since sequents are equivalence classes modulo \equiv , the relation that we get is not functional.

Definition 2.9. We say that a formula A *corresponds* to a sequent Γ when:

- either $\Gamma = A$;
- or, inductively, there exist B and C that correspond respectively to Δ and Λ such that
 - either $A = B \wp C$ and $\Gamma = [\Delta, \Lambda]$;
 - or $A = B \triangleleft C$ and $\Gamma = \langle \Delta; \Lambda \rangle$.

We shall also say that Γ corresponds to A in this case.

For instance, $a \wp a^\perp$ and $a^\perp \wp a$ both correspond to $[a, a^\perp]$, but any formula also corresponds to itself as a singleton sequent. What is important is that if a formula corresponds to a sequent, then one is provable if and only if the other is — this will clearly hold for all the systems in this paper for which this property can be stated.

2.2. Dicographs, Relation Webs and Series-Parallel Orders. In [Ret97a], Retoré presents proof nets for pomset logic as *RB-digraphs*, that is, *directed* graphs equipped with perfect matchings, extending his reformulation of MLL+mix proof nets as *undirected* RB-graphs [Ret03]. We recall these notions below.

Definition 2.10. A *digraph* $\mathcal{G} = (V_{\mathcal{G}}, R_{\mathcal{G}})$ consists of a finite set of *vertices* $V_{\mathcal{G}}$ and a set of *edges* $R_{\mathcal{G}} \subseteq V_{\mathcal{G}}^2 \setminus \{(u, u) \mid u \in V_{\mathcal{G}}\}$. A *labeled digraph* is a digraph \mathcal{G} equipped with a map $\ell: V_{\mathcal{G}} \rightarrow \mathcal{L}$ assigning each vertex v of $V_{\mathcal{G}}$ a *label* $\ell(v) \in \mathcal{L}$ in the *label set* \mathcal{L} . If \mathcal{L} is the set $\mathcal{V} \cup \mathcal{V}^\perp$ of atoms, we speak of an *atom-labeled digraph*.

Definition 2.11. An *isomorphism* between two digraphs \mathcal{G} and \mathcal{H} is a bijection on vertices $f: V_{\mathcal{G}} \xrightarrow{\sim} V_{\mathcal{H}}$ such that $f(R_{\mathcal{G}}) = R_{\mathcal{H}}$, where $f(R_{\mathcal{G}}) = \{(f(u), f(v)) \mid (u, v) \in R_{\mathcal{G}}\}$. As usual, if such an f exists, \mathcal{G} and \mathcal{H} are said to be *isomorphic*. Furthermore, if \mathcal{G} and \mathcal{H} are endowed with the vertex labelings $\ell_{\mathcal{G}}$ and $\ell_{\mathcal{H}}$ respectively, and $\ell_{\mathcal{G}} = \ell_{\mathcal{H}} \circ f$, then we say that f is an *isomorphism of labeled digraphs*.

Definition 2.12. For a given digraph $\mathcal{G} = (V_{\mathcal{G}}, R_{\mathcal{G}})$ we define the following four sets:

$$\begin{aligned} R_{\mathcal{G}}^{\otimes} &= \{(u, v) \mid (u, v) \in R_{\mathcal{G}} \text{ and } (v, u) \in R_{\mathcal{G}}\} \\ R_{\mathcal{G}}^{\triangleleft} &= \{(u, v) \mid (u, v) \in R_{\mathcal{G}} \text{ and } (v, u) \notin R_{\mathcal{G}}\} \\ R_{\mathcal{G}}^{\triangleright} &= \{(u, v) \mid (u, v) \notin R_{\mathcal{G}} \text{ and } (v, u) \in R_{\mathcal{G}}\} \\ R_{\mathcal{G}}^{\wp} &= \{(u, v) \mid (u, v) \notin R_{\mathcal{G}} \text{ and } (v, u) \notin R_{\mathcal{G}} \text{ and } u \neq v\} \end{aligned}$$

Note that these set can be seen as binary relations and as sets of edges in the graph. We will use them interchangeably in both settings.

Proposition 2.13. *For every digraph $\mathcal{G} = (V_{\mathcal{G}}, R_{\mathcal{G}})$ we have that the relations $R_{\mathcal{G}}^{\otimes}$, $R_{\mathcal{G}}^{\triangleleft}$, $R_{\mathcal{G}}^{\triangleright}$, $R_{\mathcal{G}}^{\wp}$ are pairwise disjoint and $R_{\mathcal{G}}^{\otimes} \cup R_{\mathcal{G}}^{\triangleleft} \cup R_{\mathcal{G}}^{\triangleright} \cup R_{\mathcal{G}}^{\wp} = V_{\mathcal{G}} \times V_{\mathcal{G}} \setminus \{(u, u) \mid u \in V_{\mathcal{G}}\}$. Furthermore, we have that $(u, v) \in R_{\mathcal{G}}^{\triangleleft}$ iff $(v, u) \in R_{\mathcal{G}}^{\triangleright}$.*

Remark 2.14. Conversely, a set V and four binary relations $R^{\wp}, R^{\triangleleft}, R^{\triangleright}, R^{\otimes} \subseteq V \times V$, such that

- (1) $R^{\wp}, R^{\triangleleft}, R^{\triangleright}, R^{\otimes}$ are pairwise disjoint,

⁹This can be seen as the multiplication of the monad that sends X to the generalized formulas over X .

- (2) R^\otimes and R^\circledast are symmetric,
(3) $R^\triangleleft = (R^\triangleright)^{-1}$,
(4) $R^\otimes \cup R^\triangleleft \cup R^\triangleright \cup R^\circledast = V \times V \setminus \{(v, v) \mid v \in V\}$
uniquely determines a digraph $(V, R^\triangleleft \cup R^\circledast)$.

Notation 2.15. Following [Ret97a], when drawing digraphs, we use (red/regular) arrows to denote edges in R_G^\triangleleft (and R_G^\triangleright), and arrow-free edges for R_G^\circledast . For R_G^\otimes we use no lines at all, as in the following five examples:

$$\begin{array}{ccccc}
\begin{array}{c} b \xrightarrow{\text{red}} d \\ | \\ a \xrightarrow{\text{red}} c \end{array} &
\begin{array}{c} b \xrightarrow{\text{red}} d \\ | \\ a \xrightarrow{\text{red}} c \end{array} &
\begin{array}{c} b \xrightarrow{\text{red}} d \\ \uparrow \\ a \xrightarrow{\text{red}} c \end{array} &
\begin{array}{c} b \xrightarrow{\text{red}} d \\ | \\ a \quad c \end{array} &
\begin{array}{c} b \xrightarrow{\text{red}} d \\ | \\ a \quad c \end{array} \quad (2.1)
\end{array}$$

This allows us to see (V_G, R_G^\circledast) and (V_G, R_G^\otimes) as undirected graphs.

Let us now relate (generalized) formulas and digraphs.

Definition 2.16. Let $\mathcal{G} = (V_G, R_G)$ and $\mathcal{H} = (V_H, R_H)$ be *disjoint* digraphs (i.e. $V_G \cap V_H = \emptyset$). We can define the following operations that correspond to the connectives of pomset logic and BV:

$$\begin{aligned}
\mathcal{G} \otimes \mathcal{H} &= (V_G \cup V_H, R_G \cup R_H) \\
\mathcal{G} \triangleleft \mathcal{H} &= (V_G \cup V_H, R_G \cup R_H \cup V_G \times V_H) \\
\mathcal{G} \circledast \mathcal{H} &= (V_G \cup V_H, R_G \cup R_H \cup (V_G \times V_H) \cup (V_H \times V_G))
\end{aligned}$$

Definition 2.17. The mapping $\llbracket \cdot \rrbracket$ from *linear generalized* formulas over a set X to digraphs with vertices in X is defined inductively as follows:

$$\llbracket \mathbb{I} \rrbracket = \emptyset \quad \llbracket x \rrbracket = \bullet_x \quad \llbracket A \otimes B \rrbracket = \llbracket A \rrbracket \otimes \llbracket B \rrbracket \quad \llbracket A \triangleleft B \rrbracket = \llbracket A \rrbracket \triangleleft \llbracket B \rrbracket \quad \llbracket A \circledast B \rrbracket = \llbracket A \rrbracket \circledast \llbracket B \rrbracket$$

where \emptyset is an abbreviation for the empty graph (\emptyset, \emptyset) and $\bullet_x = (\{x\}, \emptyset)$ is the unique digraph that has x as its single vertex. Note that linearity is required to fulfill the disjointness assumption of the previous definition.

Definition 2.18. If A is a generalized formula over a set X that is *not assumed to be linear*, we may translate it to a digraph as follows. Choose a *linear* generalized formula A' , a set Y and a map $\ell : Y \rightarrow X$ such that A is obtained from A' by the substitution induced by ℓ . Then the labeled digraph $(\llbracket A' \rrbracket, \ell)$ corresponds to A . We may write it $\llbracket A \rrbracket$, keeping in mind that it is only defined up to isomorphism of labeled digraphs.

Proposition 2.19. *For every linear generalized formula A we have $|A| = |V_{\llbracket A \rrbracket}|$.*

Example 2.20. The last digraph in Equation (2.1) is $\llbracket ((b \otimes a) \triangleleft d) \circledast c \rrbracket$.

The point of the map $\llbracket \cdot \rrbracket$ is to give an intrinsic representations of (generalized) formulas modulo \equiv . This is shown in [Gug07], where the result is stated in terms of “structures” and “relation webs”. The former are just formulas modulo \equiv while the latter are digraphs defined through a quadruple of relations as in Remark 2.14. Further discussion of relation webs will take place later in this subsection (Definition 2.27). For now, we give a formulation using our previous definitions.

Theorem 2.21 ([Gug07, Theorem 2.2.9]). *For any two generalized formulas A and B over some set X , we have $A \equiv B$ if and only if $\llbracket A \rrbracket$ and $\llbracket B \rrbracket$ are isomorphic labeled digraphs. If A and B are linear, this can be stated as $A \equiv B \iff \llbracket A \rrbracket = \llbracket B \rrbracket$.*

An immediate consequence of this theorem, combined with Proposition 2.8, is that we can also translate sequents into labeled digraphs.

Corollary 2.22. *The extension of the map $\llbracket \cdot \rrbracket$ to sequents by*

$$\llbracket [\Gamma, \Delta] \rrbracket = \llbracket [\Gamma] \rrbracket \otimes \llbracket [\Delta] \rrbracket \quad \text{and} \quad \llbracket \langle \Gamma; \Delta \rangle \rrbracket = \llbracket [\Gamma] \rrbracket \triangleleft \llbracket [\Delta] \rrbracket$$

is well-defined up to isomorphism of labeled digraphs. Furthermore, if the formula A corresponds to the sequent Γ according to Definition 2.9, then $\llbracket A \rrbracket$ and $\llbracket [\Gamma] \rrbracket$ are isomorphic.

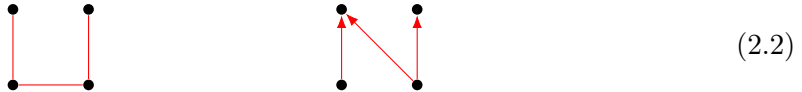
Example 2.23. The last digraph in Equation (2.1) is also $\llbracket \langle (b \otimes a; d), c \rangle \rrbracket$.

The next interesting question is how we can characterize the graphs that are translations of formulas or sequents. In fact, they form the class of *directed cographs* (which we shall abbreviate as “dicograph” as in [Ret21], cf. Definition 2.26); we refer to [BG18, Section 11.6] for a survey. It admits three characterizations that have been found independently in [BdGR97, Gug07, CP06]. They all rely on the notion of induced subgraph.

Definition 2.24. Let $\mathcal{G} = (V_{\mathcal{G}}, R_{\mathcal{G}})$ be a digraph and let $V_{\mathcal{H}} \subseteq V_{\mathcal{G}}$. The *subdigraph* of \mathcal{G} induced by $V_{\mathcal{H}}$ is $\mathcal{H} = (V_{\mathcal{H}}, R_{\mathcal{H}})$ where $R_{\mathcal{H}} = R_{\mathcal{G}} \cap (V_{\mathcal{H}} \times V_{\mathcal{H}})$. In this case we also say that \mathcal{H} is an *induced subgraph* of \mathcal{G} and denote that by $\mathcal{H} \sqsubseteq \mathcal{G}$. If additionally $V_{\mathcal{H}} \subsetneq V_{\mathcal{G}}$ then we write $\mathcal{H} \sqsubset \mathcal{G}$.

We give detailed statements for the characterizations from [BdGR97, Gug07] below, for the sake of completeness, and to show that they are very similar.

Definition 2.25. An undirected graph is \mathbf{P}_4 -free if it does not contain a \mathbf{P}_4 (shown on the left below) as induced subgraph, and a directed graph is \mathbf{N} -free if it does not contain an \mathbf{N} (shown on the right below) as induced subgraph.



Definition 2.26 ([BdGR97, Proposition 5.3]). A *dicograph* is a digraph $\mathcal{G} = (V_{\mathcal{G}}, R_{\mathcal{G}})$ such that

- (1) the undirected graph $(V_{\mathcal{G}}, R_{\mathcal{G}}^{\otimes})$ is \mathbf{P}_4 -free,
- (2) the directed graph $(V_{\mathcal{G}}, R_{\mathcal{G}}^{\triangleleft})$ is \mathbf{N} -free, and
- (3) the relation $R_{\mathcal{G}}$ is *weakly transitive*:
 - if $(u, v) \in R_{\mathcal{G}}^{\triangleleft}$ and $(v, w) \in R_{\mathcal{G}}$ then $(u, w) \in R_{\mathcal{G}}$, and
 - if $(u, v) \in R_{\mathcal{G}}$ and $(v, w) \in R_{\mathcal{G}}^{\triangleleft}$ then $(u, w) \in R_{\mathcal{G}}$.

Definition 2.27 ([Gug07, Theorem 2.2.4]). A *relation web* is a tuple $(V, R^{\otimes}, R^{\triangleleft}, R^{\triangleright}, R^{\otimes})$ obeying the four conditions in Remark 2.14 together with the following conditions:

- (5) the relations E^{\triangleleft} and E^{\triangleright} are transitive,
- (6) *Triangular property*: for any $R_1, R_2, R_3 \in \{R^{\triangleleft} \cup R^{\triangleright}, R^{\otimes}, R^{\otimes}\}$, it holds: if $(u, v) \in R_1$ and $(v, w) \in R_2$ and $(w, u) \in R_3$ then $R_1 = R_2$ or $R_2 = R_3$ or $R_3 = R_1$,
- (7) *Square property*: (V, R^{\otimes}) and (V, R^{\otimes}) are \mathbf{P}_4 -free, and (V, R^{\triangleleft}) is \mathbf{N} -free.

Theorem 2.28. *Let \mathcal{G} be a digraph. Then the following are equivalent:*

- (1) *There is a linear generalized formula A with $\mathcal{G} = \llbracket A \rrbracket$.*
- (2) *\mathcal{G} is a dicograph.*
- (3) *$(V_{\mathcal{G}}, R_{\mathcal{G}}^{\otimes}, R_{\mathcal{G}}^{\triangleleft}, R_{\mathcal{G}}^{\triangleright}, R_{\mathcal{G}}^{\otimes})$ is a relation web.*

The equivalence $1 \iff 2$ has been shown in [BdGR97, Section 5], while the equivalence $1 \iff 3$ can be found in [Gug07, Theorems 2.2.4 and 2.2.7]. A direct proof of $2 \iff 3$ is a straightforward exercise, that we encourage our readers to do by themselves.

Let us also briefly mention the characterization of dicographs from [CP06], that uses forbidden induced subgraphs as its only condition.

Theorem 2.29 ([CP06, Theorem 2]). *There exists a set F of 8 isomorphism classes of digraphs such that the class of digraphs whose induced subgraphs are not in F coincides with the class inductively generated by the operations of Definition 2.16 starting from single-vertex graphs.*

It follows from the definitions that this inductively generated class corresponds to the first item of Theorem 2.28, so this is indeed a characterization of dicographs.

Corollary 2.30. *Let $\mathcal{G} = (V_{\mathcal{G}}, R_{\mathcal{G}})$ be a dicograph. Then any induced subdigraph of \mathcal{G} is also a dicograph.*

Proof. This is because the induced subgraph relation \sqsubseteq is transitive. □

An important special case of relation webs is when $R^{\otimes} = \emptyset$.

Proposition 2.31. *Let \mathcal{G} be a digraph. The following are equivalent:*

- (1) *There is a linear generalized formula A that does not contain \otimes and such that $\mathcal{G} = \llbracket A \rrbracket$.*
- (2) *\mathcal{G} is a dicograph and $R_{\mathcal{G}}^{\otimes} = \emptyset$, i.e. all edges are unidirectional.*

Furthermore, when these conditions hold, $\mathcal{G} = (V_{\mathcal{G}}, R_{\mathcal{G}}^{\triangleleft})$. For a fixed $V_{\mathcal{G}}$, the possible values for $R_{\mathcal{G}}^{\triangleleft}$ are exactly those for which \mathcal{G} is \mathbf{N} -free and $R_{\mathcal{G}}^{\triangleleft}$ is a strict partial order. Such relations are called series-parallel orders in the literature.

Proof. Let A be a linear generalized formula. By structural induction on A , one can see that $R_{\llbracket A \rrbracket}^{\otimes} = \emptyset$ if and only if A is \otimes -free. The claimed equivalence then follows from the case of general dicographs we saw earlier. And by definition, $R_{\mathcal{G}}^{\triangleleft}$ contains all the unidirectional edges of \mathcal{G} . The characterization as \mathbf{N} -free partial orders is just the specialization of either Definition 2.26 or Definition 2.27, but it is in fact an old classical result [VTL82, Section 4] (see also [Möh89]). □

The operations \triangleleft and \triangleright on digraphs (Definition 2.16) are respectively the “series composition” and “parallel composition” of partial orders. For more on series-parallel orders, see the beginning of [BG18, Section 11.6] which points to many further references.

Remark 2.32. Sequents are \otimes -free generalized formulas modulo \equiv . By the above proposition, they are equivalent to labeled series-parallel posets up to label-preserving isomorphism, i.e. to series-parallel *partially ordered multisets (pomsets)*, as claimed in the previous subsection.

Remark 2.33. Another noteworthy special case of dicographs is those with $R_{\mathcal{G}}^{\triangleleft} = R_{\mathcal{G}}^{\triangleright} = \emptyset$; they are the digraphs of the form $\llbracket A \rrbracket$ where A does not contain \triangleleft . They correspond to a class of *undirected* graphs called *cographs*, that can be defined equivalently as \mathbf{P}_4 -free graphs (see [CLB81]). Cographs therefore provide a representation of $\mathbf{MLL} + \text{mix}$ formulas that can be used to build proof nets (analogously to the next two sections for pomset logic), see [Ret99a, Ret03].

The class of “directed cographs” has been given this name independently by [Ret99b, CP06]. This arguably attests that it is a natural generalization of cographs.

2.3. Perfect Matchings, RB-digraphs and Alternating Elementary Cycles. We have just seen representation of *formulas* as graphs. In this subsection and the next one, we recall how to graphically represent pomset logic *proofs* as well. This requires a few graph-theoretic definitions first.

Recall that an undirected graph is 1-regular if every vertex is incident to exactly one edge, and a perfect matching of some undirected graph is a 1-regular spanning subgraph (so the notion of perfect matching is usually defined relatively to some ambient graph). As a slight abuse of language, we use “perfect matching” in this paper to designate a directed counterpart to 1-regular graphs.

Definition 2.34. We say that a digraph $\mathcal{G} = (V_{\mathcal{G}}, E_{\mathcal{G}})$ is a *perfect matching* when

- (1) any vertex has exactly one outgoing edge in $E_{\mathcal{G}}$ and exactly one incoming edge in $E_{\mathcal{G}}$, i.e., for every $u \in V_{\mathcal{G}}$ there is a single pair $(v, w) \in V_{\mathcal{G}}^2$ such that $(u, v) \in E_{\mathcal{G}}$ and $(w, u) \in E_{\mathcal{G}}$, and
 - (2) all edges are bidirectional, i.e. for all $u, v \in V_{\mathcal{G}}$, we have that $(u, v) \in E_{\mathcal{G}}$ iff $(v, u) \in E_{\mathcal{G}}$.
- This means, in particular, that in the previous item, $v = w$.

Definition 2.35. An *RB-digraph* $\mathcal{G} = (V_{\mathcal{G}}, R_{\mathcal{G}}, B_{\mathcal{G}})$ is a triple where $(V_{\mathcal{G}}, R_{\mathcal{G}})$ is an arbitrary digraph and $(V_{\mathcal{G}}, B_{\mathcal{G}})$ is a perfect matching. We call the edges in $B_{\mathcal{G}}$ the *matching edges* or *B-edges*, and those in $R_{\mathcal{G}}$ the *non-matching edges* or *R-edges*. We say that \mathcal{G} is an *RB-dicograph* iff $(V_{\mathcal{G}}, R_{\mathcal{G}})$ is a dicograph.

Following Definition 2.11, we define an *isomorphism* between \mathcal{G} and another RB-digraph $\mathcal{H} = (V_{\mathcal{H}}, R_{\mathcal{H}}, B_{\mathcal{H}})$ to be a bijection $f : V_{\mathcal{G}} \xrightarrow{\sim} V_{\mathcal{H}}$ such that $f(R_{\mathcal{G}}) = R_{\mathcal{H}}$ and $f(B_{\mathcal{G}}) = B_{\mathcal{H}}$. This extends in the only sensible way to a notion of isomorphism between labeled RB-digraphs.

Notation 2.36. In all figures representing RB-digraphs, we will (following [Ret99b]) draw the matching edges **bold and blue**, while the non-matching edges will be drawn **regular and red**.

Example 2.37. Here are six examples of RB-digraphs.



In the first two, the underlying digraphs are not dicographs. But the other four examples are RB-dicographs.

Definition 2.38. Let $\mathcal{G} = (V_{\mathcal{G}}, E_{\mathcal{G}})$ be a digraph and $n \in \mathbb{N}$ with $n \geq 2$. An *elementary path* of length n in \mathcal{G} is a sequence of vertices $u_0, \dots, u_n \in V_{\mathcal{G}}$ *without repetitions* such that $(u_i, u_{i+1}) \in E_{\mathcal{G}}$ for all $i \in \{0, \dots, n-1\}$ (so the length counts the number of edges, not of vertices). An *elementary cycle* is defined in the same way except there is the single repetition $u_n = u_0$ (so the length of an elementary cycle is both its number of vertices and its number of edges).

An *alternating elementary path* (or *α -path*) in an RB-digraph $(V_{\mathcal{G}}, R_{\mathcal{G}}, B_{\mathcal{G}})$ is an elementary path u_0, \dots, u_n in the digraph $(V_{\mathcal{G}}, R_{\mathcal{G}} \cup B_{\mathcal{G}})$ such that¹⁰:

- either $(u_i, u_{i+1}) \in R_{\mathcal{G}}$ when i is odd and $(u_i, u_{i+1}) \in B_{\mathcal{G}}$ when i is even,

¹⁰One could be tempted to simplify this definition by saying that for any two consecutive edges, one is in $R_{\mathcal{G}}$ and the other in $B_{\mathcal{G}}$; the issue, however, is that $R_{\mathcal{G}}$ and $B_{\mathcal{G}}$ are not required to be disjoint.

- or $(u_i, u_{i+1}) \in R_{\mathcal{G}}$ when i is even and $(u_i, u_{i+1}) \in B_{\mathcal{G}}$ when i is odd.

An *alternating elementary cycle* (or *æ-cycle*) in $(V_{\mathcal{G}}, R_{\mathcal{G}}, B_{\mathcal{G}})$ is an elementary cycle of even length in $(V_{\mathcal{G}}, R_{\mathcal{G}} \cup B_{\mathcal{G}})$ that satisfies the above alternation condition. Morally, the parity condition ensures that the cycle also alternates between (u_{n-1}, u_n) and (u_1, u_2) , or equivalently that the “change of base points” $u'_i = u_{i+k \bmod n}$ sends æ-cycles to æ-cycles.

Example 2.39. The first and the fifth graph in Example 2.37 do not contain an æ-cycle. In all other graphs in that example, the four vertices form an æ-cycle.

The existence or non-existence of æ-cycles in RB-dicographs will play a central role in this paper. We note in passing that in the classical theory of matchings in *undirected* graphs, the absence of æ-cycles admits alternative characterizations and entails deep structural properties, which have been applied to MLL+mix proof nets [Ret03, Ngu20].

Definition 2.40. Let $u_0, \dots, u_{n-1}, u_n = u_0$ be an elementary cycle in a digraph $\mathcal{G} = (V_{\mathcal{G}}, E_{\mathcal{G}})$. An edge $(v, w) \in E_{\mathcal{G}}$ is a *chord* for this cycle when $v = u_i$ and $w = u_j$ for some $i, j \in \{0, \dots, n-1\}$ such that $i \not\equiv j+1 \pmod n$ and $j \not\equiv i+1 \pmod n$. That is, both vertices v and w occur in the cycle but the edges (v, w) and (w, v) are not part of the cycle. A cycle is *chordless* if it does not admit any chord in \mathcal{G} . A *chordless æ-cycle* in an RB-digraph $(V_{\mathcal{G}}, R_{\mathcal{G}}, B_{\mathcal{G}})$ is an æ-cycle which has no chord in the total digraph $(V_{\mathcal{G}}, R_{\mathcal{G}} \cup B_{\mathcal{G}})$. Note that since $B_{\mathcal{G}}$ is a perfect matching, if an æ-cycle admits a chord, then this chord is necessarily in $R_{\mathcal{G}} \setminus B_{\mathcal{G}}$.

Example 2.41. In Example 2.37, the æ-cycles in the second and sixth graph admit chords. The æ-cycles in the third and fourth graph are chordless.

2.4. Pomset Logic, Proof Nets and Balanced Formulas. A proof in multiplicative linear logic (MLL) is given by its conclusion (a formula or a sequent) and an *axiom linking*. This can be drawn as a graph, which consists of the formula tree (or sequent forest) with additional edges representing the axiom links, i.e., connecting those leaves of the tree which are matched by an axiom in the proof. In order to distinguish the actual proofs in the set of all such graphs, a so-called *correctness criterion* is employed, and the graphical structures that obey this criterion are called *proof nets*.

In [Ret97a], Retoré generalized this idea from the formulas/sequents of MLL to those that we introduced in Section 2.1. There exist many different equivalent correctness criteria for MLL, and one of Retoré’s main achievements was to figure out which criterion allows for a generalization to include the seq connective \triangleleft . There are in fact two such criteria, and their versions for MLL and MLL+mix has been presented in [Ret03]. They are both based on RB-graphs, and we are now going to give an exposition of their extension to RB-digraphs, as presented in [Ret97a, Ret99b, Ret21].

First, the notion of axiom linking is the same as for MLL.

Definition 2.42. A *pomset logic pre-proof* of a formula or a sequent is an involution ℓ on its set of atom occurrences such that an atom is always mapped to its dual. This ℓ is also called an *axiom linking*.

Example 2.43. The sequent $[a^\perp \wp a^\perp, a \otimes a]$ has two possible axiom linkings.

Throughout this paper, the class of formulas for which there is a single possible choice of axiom linking will play an important role.

Definition 2.44. A formula is *balanced* if every propositional variable that occurs in A occurs exactly once positively and exactly once negatively. (Equivalently, a formula is balanced when it is *linear* and its set of occurring atoms is closed under duality.) Using the correspondence of Definition 2.9, this extends to a notion of *balanced sequent*. A balanced formula A *uniquely determines* an axiom linking on A , that we denote by $\ell(A)$. Similarly, we write $\ell(\Gamma)$ for the unique axiom linking on a balanced sequent Γ .¹¹

As we have seen in the previous subsections, one can represent a formula as a dicograph, where the atom occurrences are the vertices and the edges are determined by the connectives. More traditionally, one can also associate a syntax tree to a formula. Each of these two graphical representations has an associated correctness criterion, based on \ae -cycles in RB-digraphs. We now present those two criteria, starting with dicographs.

Definition 2.45. Let Γ be a sequent and ℓ be an axiom linking for Γ . The *cographic RB-prenet* of Γ and ℓ , denoted by $\rho(\Gamma, \ell)$, is the RB-dicograph $\mathcal{G} = (V_{\mathcal{G}}, E_{\mathcal{G}}, B_{\mathcal{G}})$ where $(V_{\mathcal{G}}, E_{\mathcal{G}}) = \llbracket \Gamma \rrbracket$, and we have $(x, y) \in B_{\mathcal{G}}$ iff the atom occurrences in Γ that correspond to x and y are mapped to each other by the axiom linking ℓ .

Example 2.46. The sequent $[\langle a; a \rangle, \langle a^{\perp}; a^{\perp} \rangle]$ admits two possible axiom linkings, and the two corresponding cographic RB-prenets are the fourth and the fifth graph in Example 2.37.

Definition 2.47. We write $\llbracket A \rrbracket$ for the cographic RB-prenet $\rho(A, \ell(A))$, i.e., $\llbracket A \rrbracket = (V_{\mathcal{A}}, R_{\mathcal{A}}, B_{\mathcal{A}})$, where $(V_{\mathcal{A}}, R_{\mathcal{A}}) = \llbracket A \rrbracket$ and $B_{\mathcal{A}}$ is the matching associated to $\ell(A)$.

Proposition 2.48. *Every RB-dicograph is isomorphic to some $\llbracket A \rrbracket$ where A is a balanced formula.*

Proof. Let $\mathcal{G} = (V_{\mathcal{G}}, R_{\mathcal{G}}, B_{\mathcal{G}})$ be given. We can label the vertices in $V_{\mathcal{G}}$ with distinct atoms, such that two atoms are dual if and only if they are matched by $B_{\mathcal{G}}$. Then A exists by Theorem 2.28, since $(V_{\mathcal{G}}, R_{\mathcal{G}})$ is a dicograph. \square

Definition 2.49. A cographic RB-prenet is *correct* if it does not contain any chordless \ae -cycle. A correct cographic RB-prenet is also called a *cographic RB-net*.

Definition 2.50. A sequent Γ is *provable in pomset logic* if there is an axiom linking ℓ for Γ , such that $\rho(\Gamma, \ell)$ is correct. In that case, $\rho(\Gamma, \ell)$ (or simply ℓ) is a *pomset logic proof* of Γ .

Example 2.51. The last two graphs in Example 2.37 are pomset logic proofs for the balanced formulas $\langle a \triangleleft b \rangle \wp \langle a^{\perp} \triangleleft b^{\perp} \rangle$ and $[a \wp a^{\perp}] \otimes [b \wp b^{\perp}]$, respectively.

The following theorem says that pomset logic is a conservative extension of $\text{MLL} + \text{mix}$.

Theorem 2.52 ([Ret03, Theorem 7], see also Remark 2.33). *Let Γ be a flat sequent without any occurrence of \triangleleft , and let ℓ be an axiom linking for Γ . Then ℓ is the axiom linking of an $\text{MLL} + \text{mix}$ sequent proof iff $\rho(\Gamma, \ell)$ is correct.*

The second correctness criterion is more in the tradition of other known correctness criteria for $\text{MLL} + \text{mix}$ as it works on a structure that is directly derived from the formula trees. More precisely, we define inductively for each formula C its *RB-tree*, denoted as $\mathcal{T}_{\text{RB}}(C)$, as shown in Figure 2. Technically speaking this not a tree in the graph-theoretical sense, but we use the name as it carries the structure of the formula tree.

¹¹Balanced formulas for classical logic are discussed in detail in [Str12, Section 7].

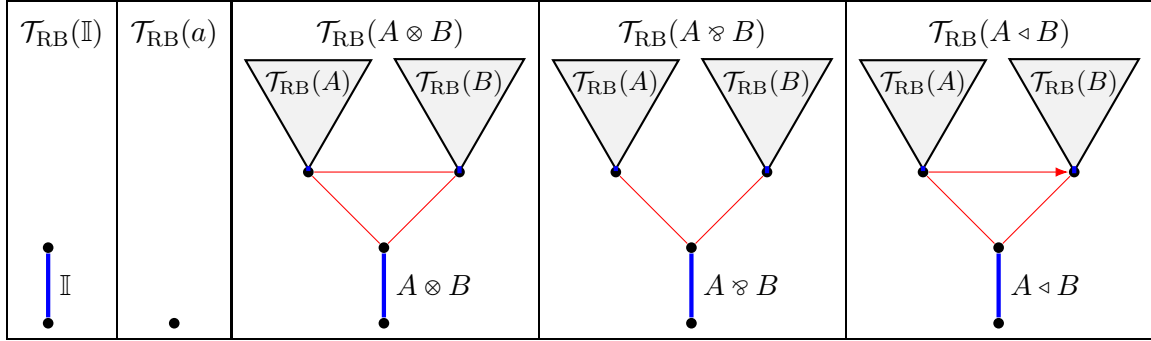


FIGURE 2. Inductive definition of RB-trees (which are not quite trees in the sense of graph theory, though they resemble the syntax trees of formulas). The root vertex is at the bottom.

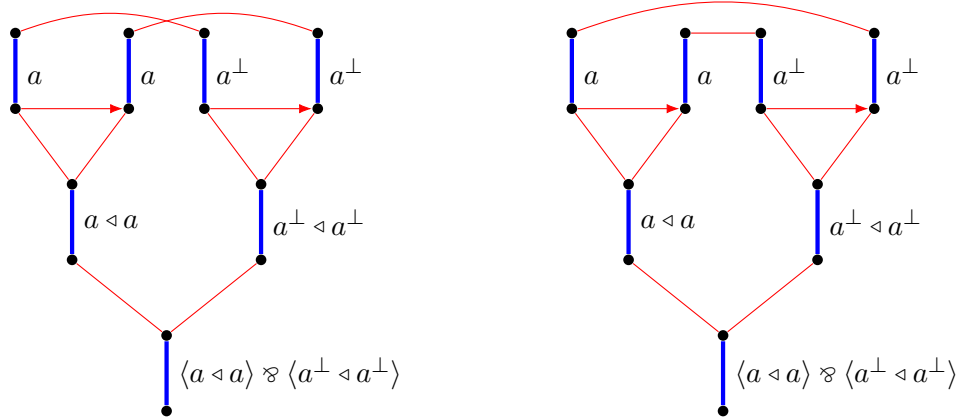


FIGURE 3. Two tree-like RB-prenets for the formula $\langle a \triangleleft a \rangle \wp \langle a^\perp \triangleleft a^\perp \rangle$. The left one is a correct proof net, while the right one contains an æ-cycle involving the 4 topmost matching edges.

If we have a sequent Γ , then $\mathcal{T}_{\text{RB}}(\Gamma)$ is obtained from the RB-trees of the formulas in Γ which are connected at the roots via the edges corresponding to the series-parallel order of the sequent structure (see the discussion following Proposition 2.31). In order to obtain an RB-digraph, we need to add the B -edges corresponding to the linking ℓ . We denote this RB-digraph by $\tau(\Gamma, \ell)$ and call it the *tree-like RB-prenet* of Γ and ℓ .

Example 2.53. The two graphs in Figure 3 show the two *tree-like RB-prenets*, corresponding to the two axiom linkings for $\langle a \triangleleft a \rangle \wp \langle a^\perp \triangleleft a^\perp \rangle$.

Below we give an alternative, more formal definition of $\tau(\Gamma, \ell)$.

Definition 2.54. Let A be a formula. We define two *unfoldings* of A , denoted by A^b and A^\sharp , to be flat sequents which are obtained as follows. For each non-atomic subformula occurrence B of A , we introduce a fresh propositional variable z_B . If A is atomic, then

$A^b = A$ and $A^\sharp = \emptyset$. Otherwise $A^b = [z_A, A^\sharp]$ and A^\sharp is defined inductively as follows:

$$\begin{aligned}
\mathbb{I}^\sharp &= z_{\mathbb{I}}^\perp \\
[B \wp C]^\sharp &= [z_{B \wp C}^\perp \otimes [z_B \wp z_C], B^\sharp, C^\sharp] \\
\langle B \triangleleft C \rangle^\sharp &= [z_{B \triangleleft C}^\perp \otimes \langle z_B \triangleleft z_C \rangle, B^\sharp, C^\sharp] \\
(B \otimes C)^\sharp &= [z_{B \otimes C}^\perp \otimes (z_B \otimes z_C), B^\sharp, C^\sharp]
\end{aligned} \tag{2.3}$$

Let Γ be a sequent, and let A_1, \dots, A_n be the formula occurrences of Γ . Then $\Gamma^b = [\Gamma_0, A_1^\sharp, \dots, A_n^\sharp]$, where Γ_0 is Γ with every A_i replaced by z_{A_i} .

Remark 2.55. It is important to note that in Definition 2.54 every subformula occurrence gets a fresh variable, in particular, in Equation (2.3) every occurrence of \mathbb{I} in A is assigned a fresh $z_{\mathbb{I}}$, and these variables have to be indexed accordingly. Then the graph $\mathcal{T}_{\text{RB}}(\Gamma)$ is in fact $[[\Gamma^b]]$ equipped with a B -edge for every fresh $z-z^\perp$ pair.

Definition 2.56. Let Γ be a sequent and ℓ an axiom linking for Γ . We define the linking ℓ^b for Γ^b to be the linking obtained from ℓ by mapping each fresh z_A to z_A^\perp , and vice versa. Then the *tree-like RB-prenet* of Γ and ℓ is defined as $\tau(\Gamma, \ell) = \rho(\Gamma^b, \ell^b)$.

The correctness criterion for tree-like RB-prenets is exactly the same as for cographic RB-prenets, except that in a tree-like RB-prenet every \ae -cycle is automatically chordless. Therefore we have the following definition.

Definition 2.57. A tree-like RB-prenet is *correct* iff it does not contain any \ae -cycle. A correct tree-like RB-prenet is also called a *tree-like RB-net*.

Example 2.58. In Figure 3, the left RB-prenet is correct, while the one on the right is not.

In [Ret99b], Retoré has shown the equivalence of the two correctness criteria.

Theorem 2.59 ([Ret99b, Theorem 7]). *For every sequent Γ and linking ℓ , we have that $\rho(\Gamma, \ell)$ is correct if and only if $\tau(\Gamma, \ell)$ is correct.*

In the remainder of this paper we will use the term *pomset logic proof net*, or short *proof net* for correct prenets of both kind, i.e., for cographic RB-nets and tree-like RB-nets. This is justified, as the two can be trivially transformed into each other in linear time.

Pomset logic proof nets can easily be extended with cut. A *cut* in a sequent is a formula of the shape $C \otimes C^\perp$. Then the cut elimination theorem can be stated as follows:

Theorem 2.60 ([Ret97a, Theorem 7]). *Let Γ be a sequent formed by the formulas $A_1, \dots, A_n, C_1 \otimes C_1^\perp, \dots, C_k \otimes C_k^\perp$, and let Γ' be obtained from Γ by removing the formulas $C_1 \otimes C_1^\perp, \dots, C_k \otimes C_k^\perp$. If there is a pomset logic proof net for Γ , then there is also one for Γ' .*

2.5. System BV and the Calculus of Structures. In [Gug99, Gug07] Guglielmi introduces *system BV*, which is a deductive system for formulas defined in Section 2.1. It is defined in the formalism called the *calculus of structures*, and it works similar to a rewriting system, modulo the equational theory defined in Figure 1.

The inference rules of *system BV* and its symmetric version *system SBV* are shown in Figure 4. These rules have to be read as rewriting rule schemes, meaning that (1) the variable a can be substituted by any atom, and the variables A, B, C, D can be substituted by any formula, and that (2) the rules can be applied inside any (positive) context.

$\text{ai}\downarrow \frac{\mathbb{I}}{a \wp a^\perp}$	$\equiv \frac{A}{B} \text{ (provided } A \equiv B)$	$\text{ai}\uparrow \frac{a \otimes a^\perp}{\mathbb{I}}$
$\text{q}\downarrow \frac{[A \wp C] \triangleleft [B \wp D]}{\langle A \triangleleft B \rangle \wp \langle C \triangleleft D \rangle}$	$\text{s} \frac{[A \wp C] \otimes B}{(A \otimes B) \wp C}$	$\text{q}\uparrow \frac{\langle A \triangleleft B \rangle \otimes \langle C \triangleleft D \rangle}{(A \otimes C) \triangleleft (B \otimes D)}$

FIGURE 4. System BV (the first two columns) and system SBV (all three columns)

More formally, an *context* $S\{\cdot\}$ is a formula which contains exactly one occurrence of the hole $\{\cdot\}$ in place of an atom:

$$S\{\cdot\} ::= \{\cdot\} \mid S\{\cdot\} \wp A \mid A \wp S\{\cdot\} \mid S\{\cdot\} \triangleleft A \mid A \triangleleft S\{\cdot\} \mid S\{\cdot\} \otimes A \mid A \otimes S\{\cdot\}$$

Given a context $S\{\cdot\}$ and a formula A , we write $S\{A\}$ to denote the formula that is obtained from $S\{\cdot\}$ by replacing the hole $\{\cdot\}$ with A .

Example 2.61. Let the context $S\{\cdot\} = \langle a \triangleleft \{\cdot\} \triangleleft b \rangle \wp (c \otimes a^\perp)$ and the formula $A = c^\perp \otimes d$ be given. Then $S\{A\} = \langle a \triangleleft (c^\perp \otimes d) \triangleleft b \rangle \wp (c \otimes a^\perp)$.

If $r \frac{A}{B}$ is an inference rule and $S\{\cdot\}$ a context, then $r \frac{S\{A\}}{AS\{B\}}$ is an instance of the rule.

A (*proof*) *system* is a set of inference rules. We write $\text{s} \parallel \delta$, or more concisely $A \vdash_{\text{s}}^\delta B$, if

there is a derivation from A to B using only rules from the system S , and that derivation is named δ . If in that situation $A = \mathbb{I}$, then we write it as $\text{s} \parallel \delta$ or simply as $\vdash_{\text{s}}^\delta B$ and call δ a *proof* of B . In this case we say that B is *provable* S .

Figure 5 shows an example for a proof in BV.

We now recall some basic properties of BV and SBV. First, observe that the rules $\text{ai}\downarrow$ (called *atomic interaction down* or *axiom*) and $\text{ai}\uparrow$ (called *atomic interaction up* or *cut*) are in atomic form. Their general forms

$$\text{i}\downarrow \frac{\mathbb{I}}{A \wp A^\perp} \quad \text{and} \quad \text{i}\uparrow \frac{A \otimes A^\perp}{\mathbb{I}} \quad (2.4)$$

are derivable.

Definition 2.62. An inference rule r is *derivable* in a system S iff for every instance $r \frac{A}{B}$ there is a derivation $A \vdash_{\text{S}} B$. An inference rule r is *admissible* for a system S iff for every proof $\vdash_{\text{S} \cup \{r\}} A$ there is a proof $\vdash_{\text{S}} B$.

Proposition 2.63. *The rule $\text{i}\downarrow$ is derivable in BV, and the rule $\text{i}\uparrow$ is derivable in SBV.*

The proof is standard and can be found in many papers (e.g., [Gug07, GS01, TS19]). We are now going to state the cut elimination property for BV.

Definition 2.64. Two systems S_1 and S_2 are *equivalent* if they prove the same formulas.

Theorem 2.65 ([GS01, Gug07]). *Systems BV and SBV are equivalent.*

$$\begin{array}{c}
\text{ai}\downarrow \frac{\text{II}}{e^\perp \wp e} \\
\text{ai}\downarrow \frac{[e^\perp \wp e] \triangleleft [b^\perp \wp b]}{[e^\perp \wp e] \triangleleft [b^\perp \wp b]} \\
\text{q}\downarrow \frac{\langle e^\perp \triangleleft b^\perp \rangle \wp \langle e \triangleleft b \rangle}{\langle e^\perp \triangleleft b^\perp \rangle \wp \langle e \triangleleft b \rangle} \\
\text{q}\downarrow \frac{b^\perp \wp e^\perp \wp \langle e \triangleleft b \rangle}{b^\perp \wp e^\perp \wp \langle e \triangleleft b \rangle} \\
\text{ai}\downarrow \frac{\langle [c \wp c^\perp] \triangleleft [b^\perp \wp e^\perp] \rangle \wp \langle e \triangleleft b \rangle}{\langle [c \wp c^\perp] \triangleleft [b^\perp \wp e^\perp] \rangle \wp \langle e \triangleleft b \rangle} \\
\text{q}\downarrow \frac{\langle c \triangleleft b^\perp \rangle \wp \langle c^\perp \triangleleft e^\perp \rangle \wp \langle e \triangleleft b \rangle}{\langle c \triangleleft b^\perp \rangle \wp \langle c^\perp \triangleleft e^\perp \rangle \wp \langle e \triangleleft b \rangle} \\
\text{ai}\downarrow \frac{([a \wp a^\perp] \otimes \langle c \triangleleft b^\perp \rangle) \wp \langle c^\perp \triangleleft e^\perp \rangle \wp \langle e \triangleleft b \rangle}{([a \wp a^\perp] \otimes \langle c \triangleleft b^\perp \rangle) \wp \langle c^\perp \triangleleft e^\perp \rangle \wp \langle e \triangleleft b \rangle} \\
\text{s} \frac{(a \otimes \langle c \triangleleft b^\perp \rangle) \wp a^\perp \wp \langle c^\perp \triangleleft e^\perp \rangle \wp \langle e \triangleleft b \rangle}{(a \otimes \langle c \triangleleft b^\perp \rangle) \wp a^\perp \wp \langle c^\perp \triangleleft e^\perp \rangle \wp \langle e \triangleleft b \rangle} \\
\text{ai}\downarrow \frac{(a \otimes \langle c \triangleleft b^\perp \rangle) \wp \langle [a^\perp \wp \langle c^\perp \triangleleft e^\perp \rangle] \triangleleft [f \wp f^\perp] \rangle \wp \langle e \triangleleft b \rangle}{(a \otimes \langle c \triangleleft b^\perp \rangle) \wp \langle [a^\perp \wp \langle c^\perp \triangleleft e^\perp \rangle] \triangleleft [f \wp f^\perp] \rangle \wp \langle e \triangleleft b \rangle} \\
\text{q}\downarrow \frac{(a \otimes \langle c \triangleleft b^\perp \rangle) \wp \langle a^\perp \triangleleft f \rangle \wp \langle c^\perp \triangleleft e^\perp \triangleleft f^\perp \rangle \wp \langle e \triangleleft b \rangle}{(a \otimes \langle c \triangleleft b^\perp \rangle) \wp \langle a^\perp \triangleleft f \rangle \wp \langle c^\perp \triangleleft e^\perp \triangleleft f^\perp \rangle \wp \langle e \triangleleft b \rangle} \\
\text{ai}\downarrow \frac{(a \otimes \langle c \triangleleft b^\perp \rangle) \wp \langle a^\perp \triangleleft f \rangle \wp \langle c^\perp \triangleleft ([d^\perp \wp d] \otimes \langle e^\perp \triangleleft f^\perp \rangle) \rangle \wp \langle e \triangleleft b \rangle}{(a \otimes \langle c \triangleleft b^\perp \rangle) \wp \langle a^\perp \triangleleft f \rangle \wp \langle c^\perp \triangleleft ([d^\perp \wp d] \otimes \langle e^\perp \triangleleft f^\perp \rangle) \rangle \wp \langle e \triangleleft b \rangle} \\
\text{s} \frac{(a \otimes \langle c \triangleleft b^\perp \rangle) \wp \langle a^\perp \triangleleft f \rangle \wp \langle c^\perp \triangleleft [d^\perp \wp (d \otimes \langle e^\perp \triangleleft f^\perp \rangle)] \rangle \wp \langle e \triangleleft b \rangle}{(a \otimes \langle c \triangleleft b^\perp \rangle) \wp \langle a^\perp \triangleleft f \rangle \wp \langle c^\perp \triangleleft [d^\perp \wp (d \otimes \langle e^\perp \triangleleft f^\perp \rangle)] \rangle \wp \langle e \triangleleft b \rangle} \\
\text{q}\downarrow \frac{(a \otimes \langle c \triangleleft b^\perp \rangle) \wp \langle a^\perp \triangleleft f \rangle \wp \langle c^\perp \triangleleft d^\perp \rangle \wp (d \otimes \langle e^\perp \triangleleft f^\perp \rangle) \wp \langle e \triangleleft b \rangle}{(a \otimes \langle c \triangleleft b^\perp \rangle) \wp \langle a^\perp \triangleleft f \rangle \wp \langle c^\perp \triangleleft d^\perp \rangle \wp (d \otimes \langle e^\perp \triangleleft f^\perp \rangle) \wp \langle e \triangleleft b \rangle}
\end{array}$$

FIGURE 5. Example of a proof in BV. Instances of \equiv are left implicit and are omitted. We highlight the principal subformula in the conclusion of each rule instance.

A proof of this result can be found in [Gug07] and in [Str03b].

Theorem 2.66 ([GS01, Gug07]). *The general cut rule \uparrow is admissible for BV.*

This is an immediate corollary of Proposition 2.63 and Theorem 2.65. Finally, the relation between BV and SBV can be strengthened to the following statement:

Corollary 2.67. *For any two formulas A and B , we have $\vdash_{\text{BV}} A^\perp \wp B$ iff $A \vdash_{\text{SBV}} B$.*

2.6. Unit-Free Versions of BV and SBV. One of the main reasons to study cut-free systems is to have a deductive system that is suitable for doing proof search. However, due to the versatility of the unit II in formulas, proof search in plain BV as it is shown in Figure 4 is not feasible. In order to reduce the non-determinism in BV, Kahramanoğulları proposed in [Kah04] a unit-free version of BV which is better suited for proof search. As this system is also easier to handle for some results we show in this paper, we introduce BVu below. For didactic reasons, we also introduce its symmetric version SBVu.

The formulas for BVu are the same as defined in Section 2.1, except that we do not allow any occurrence of the unit II . This means that we have to restrict the equivalence defined in Figure 1 to the unit-free formulas. We define the relation \equiv' to be the smallest congruence generated by

$$\begin{array}{lcl}
A \otimes (B \otimes C) & \equiv' & (A \otimes B) \otimes C \\
A \wp [B \wp C] & \equiv' & [A \wp B] \wp C \\
A \triangleleft \langle B \triangleleft C \rangle & \equiv' & \langle A \triangleleft B \rangle \triangleleft C \\
A \otimes B & \equiv' & B \otimes A \\
A \wp B & \equiv' & B \wp A
\end{array} \tag{2.5}$$

The inference rules for BVu and SBVu are then shown in Figure 6. Note that the rule $\text{ai}\downarrow$ has no premise. It is an axiom that is used exactly once in a *proof* which in BVu or SBVu

$\text{ai}^\circ\downarrow \frac{}{a \wp a^\perp}$	$\text{ai}^\circ\downarrow \frac{B}{[a \wp a^\perp] \otimes B} \equiv' \frac{A}{B} \text{ (provided } A \equiv' B)$	$\text{ai}^\circ\uparrow \frac{(a \otimes a^\perp) \wp B}{B}$	$\text{ai}^\circ\uparrow \frac{a \otimes a^\perp}{}$	
$\text{ai}_L^\circ\downarrow \frac{B}{[a \wp a^\perp] \triangleleft B}$	$\text{ai}_R^\circ\downarrow \frac{B}{B \triangleleft [a \wp a^\perp]}$	$\text{ai}_L^\circ\uparrow \frac{(a \otimes a^\perp) \triangleleft B}{B}$	$\text{ai}_R^\circ\uparrow \frac{B \triangleleft (a \otimes a^\perp)}{B}$	
$\text{q}_3^\downarrow \frac{[A \wp C] \triangleleft B}{\langle A \triangleleft B \rangle \wp C}$	$\text{q}_3^\downarrow \frac{A \triangleleft [B \wp C]}{\langle A \triangleleft B \rangle \wp C}$	$s_3 \frac{[A \wp C] \otimes B}{(A \otimes B) \wp C}$	$\text{q}_3^\uparrow \frac{\langle A \triangleleft B \rangle \otimes C}{(A \otimes C) \triangleleft B}$	$\text{q}_3^\uparrow \frac{\langle A \triangleleft B \rangle \otimes C}{A \triangleleft (B \otimes C)}$
$\text{q}_4^\downarrow \frac{[A \wp C] \triangleleft [B \wp D]}{\langle A \triangleleft B \rangle \wp \langle C \triangleleft D \rangle}$	$\text{q}_2^\downarrow \frac{A \triangleleft B}{A \wp B}$	$s_2 \frac{A \otimes B}{A \wp B}$	$\text{q}_2^\uparrow \frac{A \otimes B}{A \triangleleft B}$	$\text{q}_4^\uparrow \frac{\langle A \triangleleft B \rangle \otimes \langle C \triangleleft D \rangle}{(A \otimes C) \triangleleft (B \otimes D)}$

FIGURE 6. System BVu (first three columns) and system SBVu (all five columns)

is a derivation without premise (as the unit \mathbb{I} is not present and cannot take this role). Likewise, the rule $\text{ai}^\circ\uparrow$ has no conclusion. It is used exactly once in a *refutation*, which is a derivation with empty conclusion.

We have the following immediate results.

Proposition 2.68. *Let A and B be unit-free formulas. We have $A \vdash_{\text{SBVu}} B$ iff $A \vdash_{\text{SBV}} B$.*

Proof. If we have a derivation $A \vdash_{\text{SBVu}} B$, then we immediately have a derivation $A \vdash_{\text{SBV}} B$, as every rule in SBVu (except for $\text{ai}^\circ\downarrow$ and $\text{ai}^\circ\uparrow$) is derivable in SBV. Conversely, assume we have a derivation $A \vdash_{\text{SBV}}^\delta B$. Then, in δ , the unit \mathbb{I} can occur. Let δ' be obtained from δ by deleting the unit \mathbb{I} everywhere. Then every instance of the rule \equiv becomes an instance of \equiv' ; every instance of $\text{q}\downarrow$ becomes an instance of q_2^\downarrow or $\text{q}_3^L\downarrow$ or $\text{q}_3^R\downarrow$ or q_4^\downarrow or trivial (i.e., premise and conclusion of the rule instance become equal); and similarly for s and $\text{q}\uparrow$. However, an instance of $\text{ai}\downarrow$ can become an instance of $\text{ai}^\circ\downarrow$ or $\text{ai}_L^\circ\downarrow$ or $\text{ai}_R^\circ\downarrow$ (which are in SBVu), or $\text{ai}^\circ\downarrow$ which is shown on the left below. Similarly, an instance of $\text{ai}\uparrow$ can become an instance of $\text{ai}^\circ\uparrow$ or $\text{ai}_L^\circ\uparrow$, or $\text{ai}_R^\circ\uparrow$, or $\text{ai}^\circ\uparrow$ which is shown on the right below:

$$\text{ai}^\circ\downarrow \frac{B}{a \wp a^\perp \wp B} \qquad \text{ai}^\circ\uparrow \frac{a \otimes a^\perp \otimes B}{B} \qquad (2.6)$$

These rules are not in SBVu, but they can be derived with $\{\text{ai}^\circ\downarrow, s_2\}$ and $\{\text{ai}^\circ\uparrow, s_2\}$, respectively. \square

Proposition 2.69 ([Kah04]). *The systems BVu and BV are equivalent.*

Proof. First, if we have a proof $\vdash_{\text{BVu}} A$ then we can simply replace the top instance of $\text{ai}^\circ\downarrow$ by $\text{ai}\downarrow$ and have a proof of BV. Conversely, a proof in BV can be transformed into a proof of BVu replacing the to axiom $\text{ai}^\circ\downarrow$ by a $\text{ai}\downarrow$ and then follow the same procedure as in the previous proof. \square

From these propositions we can immediately obtain the cut elimination results for BVu via the corresponding results for BV:

Corollary 2.70. *The systems BVu and SBVu are equivalent.*

Corollary 2.71. *For any two unit-free formulas we have that $\vdash_{\text{BVu}} A^\perp \wp B$ iff $A \vdash_{\text{SBVu}} B$.*

Corollary 2.72. *The general cut rules*

$$i^{\otimes\uparrow} \frac{(A \otimes A^\perp) \wp B}{B} \quad i^{\wp\uparrow} \frac{A \otimes A^\perp \otimes B}{B} \quad i^{\triangleleft\uparrow} \frac{(A \otimes A^\perp) \triangleleft B}{B} \quad i^{\triangleleft\uparrow} \frac{B \triangleleft (A \otimes A^\perp)}{B} \quad (2.7)$$

are admissible for BVu .

Remark 2.73. Our version of BVu is slightly different from the one by Kahramanoğulları [Kah04]. In [Kah04] the rule s_2 is absent, and instead the rule $ai^{\wp\uparrow}$ shown in Equation (2.6) is part of the system. We chose this variation because it is better suited for the results of this paper (e.g. Theorem 2.77 and the proofs in Section 3). But it is easy to see that the two variants of BVu are equivalent: first, as we have mentioned above, the rule $ai^{\otimes\uparrow}$ is derivable in $\{ai^{\otimes\downarrow}, s_2\}$, and second, the rule s_2 is admissible if $ai^{\otimes\uparrow}$ is present. This can be seen by an easy induction on the size of the derivation. However, note that the same trick does not work for the rule $q_2\downarrow$. This rule cannot be shown admissible, as the formula $\langle a \triangleleft [b \wp c] \rangle \wp \langle [a^\perp \wp b^\perp] \triangleleft c^\perp \rangle$ is not provable in BVu without $q_2\downarrow$.

Remark 2.74. The logical rules of SBVu , i.e., the bottom two lines in the Figure 6, have already been studied by Retoré in [Ret99b], as a rewrite system on digraphs to generate theorems of pomset logic. We will study the relation between pomset logic and BV/BVu in the next section.

In some sections of this paper we also need a variant of BVu that we call $\text{BV}\hat{u}$ and that is obtained from BVu by restricting rules $q_2\downarrow$ and s_2 to cases where neither A nor B has a \wp as main connective, i.e., we replace $q_2\downarrow$ and s_2 by $\hat{q}_2\downarrow$ and \hat{s}_2 , respectively:

$$\hat{q}_2\downarrow \frac{A \triangleleft B}{A \wp B} \quad \hat{s}_2 \frac{A \otimes B}{A \wp B} \quad \text{where } A \not\equiv C \wp D \text{ and } B \not\equiv C \wp D \text{ for any formulas } C \text{ and } D. \quad (2.8)$$

and similarly, by restricting the rules $q_3^L\downarrow$, $q_3^R\downarrow$, and s_3 to cases where C does not have a \wp as main connective, i.e., these three rules are replaced by $\hat{q}_3^L\downarrow$, $\hat{q}_3^R\downarrow$, and \hat{s}_3 , respectively:

$$\hat{q}_3^L\downarrow \frac{[A \wp C] \triangleleft B}{\langle A \triangleleft B \rangle \wp C} \quad \hat{q}_3^R\downarrow \frac{A \triangleleft [B \wp C]}{\langle A \triangleleft B \rangle \wp C} \quad \hat{s}_3 \frac{[A \wp C] \otimes B}{(A \otimes B) \wp C} \quad \text{where } C \not\equiv D \wp E \text{ for any formulas } D \text{ and } E. \quad (2.9)$$

Proposition 2.75. *The systems BVu and $\text{BV}\hat{u}$ are equivalent.*

Proof. Any derivation in $\text{BV}\hat{u}$ is also a derivation in BVu . Conversely, the rules $q_2\downarrow$ and s_2 and s_3 are derivable with $\{\hat{q}_2\downarrow, \hat{q}_3^L\downarrow, \hat{q}_3^R\downarrow, \equiv\}$ and $\{\hat{s}_2, \hat{s}_3, \equiv\}$ and $\{\hat{s}_3, \equiv\}$, respectively, as shown below:

$$\begin{array}{l} \hat{q}_3^R\downarrow \frac{[A' \wp A''] \triangleleft [B' \wp B'']}{\langle [A' \wp A''] \triangleleft B' \rangle \wp B''} \\ \hat{q}_3^L\downarrow \frac{\langle A' \triangleleft B' \rangle \wp A'' \wp B''}{A' \wp B' \wp A'' \wp B''} \\ \hat{q}_2\downarrow \frac{A' \wp B' \wp A'' \wp B''}{[A' \wp A''] \wp [B' \wp B'']} \\ \equiv' \end{array} \quad \begin{array}{l} \equiv', \hat{s}_3, \equiv' \frac{[A' \wp A''] \otimes [B' \wp B'']}{([A' \wp A''] \otimes B') \wp B''} \\ \hat{s}_3 \frac{(A' \otimes B') \wp A'' \wp B''}{A' \wp B' \wp A'' \wp B''} \\ \hat{s}_2 \frac{A' \wp B' \wp A'' \wp B''}{[A' \wp A''] \wp [B' \wp B'']} \\ \equiv' \end{array} \quad \begin{array}{l} \equiv' \frac{[A \wp [C', C'']] \otimes B}{[[A \wp C'] \wp C''] \otimes B} \\ \hat{s}_3 \frac{([A \wp C'] \otimes B) \wp C''}{[(A \otimes B) \wp C'] \wp C''} \\ \hat{s}_3 \frac{[(A \otimes B) \wp C'] \wp C''}{(A \otimes B) \wp [C' \wp C'']} \\ \equiv' \end{array}$$

and similarly, the rules $q_3^L\downarrow$ and $q_3^R\downarrow$ are derivable in $\{\hat{q}_3^L\downarrow, \equiv\}$ and $\{\hat{q}_3^R\downarrow, \equiv\}$, respectively. \square

2.7. SBVu and Dicograph Inclusions. To wrap up these long preliminaries, we recall here some useful results that also provide some motivation for the rules of SBV and SBVu. The starting observation is that the non-interaction rules preserve the atoms of a formula, so they can be seen as digraph rewriting rules preserving the set of vertices. The following results, due to B echet, de Groote and Retor e [BdGR97], elucidate the combinatorial meaning of this rewriting system.

First, let us consider the case of unit-free formulas *without the tensor connective* \otimes . Recall that according to Proposition 2.31, such formulas modulo \equiv' correspond to *series-parallel orders* on their atom occurrences.

Theorem 2.76 ([BdGR97, Propositions 3.2 and 4.1], reformulated). *Let A and B be two unit-free and tensor-free linear generalized formulas over some set X . Then the inclusion of edges $R_{\llbracket A \rrbracket} \supseteq R_{\llbracket B \rrbracket}$ holds if and only if $A \vdash B$ in the fragment of BVu where the interaction rules and the rules involving tensors have been excluded: $\{\equiv', \mathfrak{q}_2\downarrow, \mathfrak{q}_3^L\downarrow, \mathfrak{q}_3^R\downarrow, \mathfrak{q}_4\downarrow\}$.*

Equivalently, this is also the non-interaction non-tensor fragment of SBVu, since all the rules of SBVu that are not in BVu contain tensors.

Proof. Let us explain how to connect this reformulation to the original statement in [BdGR97]. In the latter, the inclusion $R_{\llbracket A \rrbracket} \supseteq R_{\llbracket B \rrbracket}$ is characterized by a rewriting system on series-parallel orders whose rules are listed in [BdGR97, Definition 3.1]. Observe that, among those rules:

- (a), (b), (c) and (d) correspond to $\mathfrak{q}_2\downarrow, \mathfrak{q}_3^R\downarrow, \mathfrak{q}_3^L\downarrow, \mathfrak{q}_4\downarrow$ respectively;
- (e) and (h) express the reflexivity and transitivity of the rewriting relation, which corresponds to the fact that a derivation is a sequence of zero, one or more inference rules;
- the remaining rules, namely (f) and (g), express the contextual closure of the rules, whose counterpart in our setting is deep inference, i.e. the possibility of instantiating a rule in a context $S\{\cdot\}$.

There remains a subtlety: we must use the \equiv' rule to turn formulas into equivalent ones on which other rules may be applied, whereas this is left implicit in the setting of [BdGR97] (since \equiv' on formulas corresponds to equality of dicographs). The fact that \equiv' suffices is due to the fact that “the algebraic representation of any [series-parallel] order is unique modulo the associativity of \triangleleft, \wp and the commutativity of \wp ” (a quote from [BdGR97, §2] where we adapted the notations for the connectives). \square

Next, we turn to unit-free formulas that may contain tensors; their associated graphs are all dicographs. One would then expect the non-interaction fragment of SBVu to characterize dicograph inclusion. However, this is not quite so, since one rule must be added.

Theorem 2.77 (reformulation of [BdGR97, §5]). *Let A and B be two unit-free linear generalized formulas over a set X . Then the inclusion of edges $R_{\llbracket A \rrbracket} \supseteq R_{\llbracket B \rrbracket}$ holds if and only if $A \vdash B$ in the fragment of SBVu without the interaction rules, plus the following additional weak switch rule:*

$$\text{ws} \frac{[A \wp B] \otimes [C \wp D]}{(A \otimes C) \wp (B \otimes D)}$$

Remark 2.78. In [BdGR97, §5], it is suggested that the proof of Theorem 2.76 can be carried over to give a proof of Theorem 2.77. This is not immediately true. Consider for example $A = (a \otimes b) \triangleleft c$ and $B = a \triangleleft [b \wp c]$. Clearly $R_{\llbracket A \rrbracket} \supseteq R_{\llbracket B \rrbracket}$, so there must be a derivation from A to B . To construct this derivation, the proof in [BdGR97] proceeds by

induction on $|V_{\llbracket A \rrbracket}| = |V_{\llbracket B \rrbracket}|$ and makes a case analysis on the main connectives of A and B . In our case it is \triangleleft for both. But the argument that works for series-parallel orders does not work for dicographs in general. In our example, we have to go through $a \triangleleft b \triangleleft c$. However with some adjustments, the proof does go through. Since this paper is already quite long, we refrain from giving the details, as they are quite straightforward, once the aforementioned problem is observed.

Remark 2.79. We shall see in Section 3.1 that the rules of BV preserve pomset logic correctness. As Retoré noticed [Ret99b, §5], this is *not* the case for the weak switch rule, and this provides one justification for excluding it from SBV .

Another argument, which is more intrinsic to BV , is that $BV + ws$ does not admit cut-elimination, or equivalently, that $SBV + ws$ is not conservative over $BV + ws$. The exclusion of the weak switch for this reason is explained as a deliberate design choice by Guglielmi in the discussion concerning “conservation laws” at the beginning of [Gug07, §3].

Remark 2.80. By analogy with the previous subsection, we could show that the non-interaction fragment of $SBVu + ws$ is equivalent over unit-free formulas to $\{\equiv, q\downarrow, q\uparrow, ws\}$ with units (indeed, the usual switch is an instance of the weak switch when one of the formulas is set to a unit). Then, Theorem 2.77 above becomes equivalent to [Gug07, Conjecture 3.3.3], which states that $\{\equiv, q\downarrow, q\uparrow, ws\}$ characterizes dicograph inclusion. In other words, that conjecture had been proved before it was stated.

3. COMPARING BV AND POMSET LOGIC

In this section we investigate the relation between the two logics. We have already seen in Section 2.1 that every formula uniquely determines a dicograph. Furthermore, by inspecting the rules of BV in Figure 4, one can see that the rule \equiv does not change that dicograph, and that the rules s and $q\downarrow$ only change the set of edges but not the set of vertices of the corresponding dicograph. Additionally, every instance of $ai\downarrow$ removes one pair of dual atoms, and in a proof of BV , every atom occurring in the conclusion has to be removed by exactly one instance of $ai\downarrow$ in the proof.

This means that every BV proof δ uniquely determines an axiom linking $\ell(\delta)$ for its conclusion, and hence by definition also a pomset logic pre-proof, which in turn, by Definition 2.45, determines a cographic RB-prenet.

In Section 3.1 we are going to show that every cographic RB-prenet that is obtained from a BV proof in such a way is indeed correct, and therefore every theorem of BV is also a theorem of pomset logic.

Then, in Section 3.2 we show that the converse does not hold, i.e., there are theorems of pomset logic that are not theorems of BV . We do this by presenting a formula that is provable in pomset logic but not in BV .

3.1. BV is Contained in Pomset Logic. In this section we do not only show that every theorem of BV is also a theorem of pomset logic, but also that every proof in BV uniquely determines a pomset logic proof net with the same conclusion.

The proof that we present here uses the basic idea from [Str03b] that has also been used in [Str03a]. In [Ret99b], Retoré presents an alternative method.

To begin, let δ be a BV proof of a formula A . We denote by $\llbracket \delta \rrbracket = \rho(A, \ell(\delta))$ the cographic RB-prenet generated from δ as described above (see Definition 2.45). Then the main result of this section is the following.

Theorem 3.1. *For every BV proof δ , the cographic RB-prenet $\llbracket \delta \rrbracket$ is correct.*

In order to prove it, observe first that every RB-dicograph uniquely determines a balanced formula, up to renaming of variables and equivalence under \equiv . This gives us immediately the following proposition.

Proposition 3.2. *Let δ be a proof in BV. Then there is a balanced formula A , that is provable in BV and such that $\llbracket A \rrbracket$ and $\llbracket \delta \rrbracket$ are isomorphic.*

Proof. Let B be the conclusion of δ . Then A is obtained from B by renaming all variable occurrences such that the result is balanced and the linking is preserved. \square

Definition 3.3. Let A and B be generalized formulas over a set X . We call B a *pseudo-subformula* of A , written as $B \sqsubseteq A$, if it is equivalent under \equiv to some A' that can be obtained from A by replacing some occurrences of elements of X (in the case of usual formulas, some atom occurrences) in A by \mathbb{I} . If $B \sqsubseteq A$ and $B \not\equiv A$, then we say that B is a *proper pseudo-subformula* of A , and write it as $B \sqsubset A$.

Example 3.4. We have the pseudo-subformula relation

$$\langle a \triangleleft d \rangle \wp (b \otimes b) \sqsubset \langle (a \otimes b) \triangleleft d \triangleleft e \rangle \wp (b \otimes [(e \otimes f) \wp \langle a \triangleleft b \rangle])$$

coming from the equivalence

$$\langle a \triangleleft d \rangle \wp (b \otimes b) \equiv \langle (a \otimes \mathbb{I}) \triangleleft d \triangleleft \mathbb{I} \rangle \wp (b \otimes [(\mathbb{I} \otimes \mathbb{I}) \wp \langle \mathbb{I} \triangleleft b \rangle])$$

The following proposition explains our choice to denote both pseudo-subformulas and induced subgraphs (Definition 2.24) by \sqsubseteq .

Proposition 3.5. *Let A and B be linear generalized formulas (this is the case, for instance, when A and B are balanced usual formulas). We have that $B \sqsubseteq A$ if and only if $\llbracket B \rrbracket \sqsubseteq \llbracket A \rrbracket$, and $B \sqsubset A$ if and only if $\llbracket B \rrbracket \sqsubset \llbracket A \rrbracket$.*

Proof. Let $X = V_{\llbracket A \rrbracket}$. By definition, the induced subgraphs $\mathcal{H} \sqsubseteq \llbracket A \rrbracket$ are in bijection with the subsets $Y \subseteq X$ via $V_{\mathcal{H}} = Y$. Let $A_{\upharpoonright Y}$ be the formula obtained from A by replacing every $x \in X \setminus Y$ by \mathbb{I} ; from the inductive definition of $\llbracket \cdot \rrbracket$ one can check that $\llbracket A_{\upharpoonright Y} \rrbracket = \mathcal{H}$. To conclude, apply Theorem 2.21 to handle the equivalence \equiv that appears in the definition of pseudo-subformula. \square

Lemma 3.6. *Let A be a balanced formula (Definition 2.44) and B be a balanced pseudo-subformula of A . If A is provable in BV, then so is B .*

Proof. Let δ be the proof of A in BV, and let δ' be obtained by replacing all atoms that do not occur in B in every line of δ by \mathbb{I} . Then δ' is a valid derivation of B in BV. \square

Definition 3.7. Let H be a balanced formula and $\mathcal{H} = \llbracket H \rrbracket$. We say that H is a (*balanced*) *cycle* when the RB-digraph \mathcal{H} admits a *chordless* \mathfrak{a} -cycle that visits all vertices¹² and either $|V_{\mathcal{H}}| = 2$ or $R_{\mathcal{H}} \cap B_{\mathcal{H}} = \emptyset$.

¹²An elementary cycle in a digraph that visits all vertices is generally called a *Hamiltonian cycle*.

Proposition 3.8. *A formula H is a balanced cycle if and only if there are pairwise distinct atoms a_1, \dots, a_n for some $n \geq 1$, such that $H \equiv L_1 \wp L_2 \wp \dots \wp L_n$, where $L_1 = a_n^\perp \otimes a_1$ or $L_1 = a_n^\perp \triangleleft a_1$, and for every $i \in \{2, \dots, n\}$ we have $L_i = a_{i-1}^\perp \otimes a_i$ or $L_i = a_{i-1}^\perp \triangleleft a_i$.*

Proof. This follows almost immediately from the definitions. \square

Definition 3.9. We say that a balanced formula A contains a cycle if it has a pseudo-subformula $B \sqsubseteq A$ that is a cycle (or equivalently, if $\llbracket A \rrbracket$ contains a chordless æ-cycle).

We are now ready to state and prove the central lemma to this section.

Lemma 3.10. *Let $r \frac{Q}{P}$ be an instance of an inference rule in $\text{BV}\hat{u}$. If P is a balanced cycle then Q contains a cycle. If $r \neq \equiv'$ then the size of the cycle in Q is strictly smaller than $|P|$.*

Proof. By Proposition 3.8 we have that $P \equiv' L_1 \wp L_2 \wp \dots \wp L_n$, where $L_1 = a_n^\perp \otimes a_1$ or $L_1 = a_n^\perp \triangleleft a_1$, and for every $i \in \{2, \dots, n\}$ we have $L_i = a_{i-1}^\perp \otimes a_i$ or $L_i = a_{i-1}^\perp \triangleleft a_i$, with all a_i being pairwise distinct. We proceed by case analysis on the rule r . First observe that by Proposition 3.8 the rules $\text{ai}^\otimes\downarrow$, $\text{ai}^\triangleleft\downarrow$, $\text{ai}^\triangleleft\downarrow$ cannot be applied to P (seen bottom up), and if $r = \equiv'$, then Q trivially contains a cycle, whose size is equal to $|P|$. Now assume r is

- $\text{q4}\downarrow \frac{[A \wp C] \triangleleft [B \wp D]}{\langle A \triangleleft B \rangle \wp \langle C \triangleleft D \rangle}$: Without loss of generality, assume that $A = a_n^\perp$ and $B = a_1$ and $C = a_{i-1}^\perp$ and $D = a_i$ for some $i \in \{2, \dots, n\}$. Then

$$Q \equiv' \langle [a_n^\perp \wp a_{i-1}^\perp] \triangleleft [a_1 \wp a_i] \rangle \wp L_2 \wp \dots \wp L_{i-1} \wp L_{i+1} \wp \dots \wp L_n$$

which contains the cycle $\langle a_n^\perp \triangleleft a_i \rangle \wp L_{i+1} \wp \dots \wp L_n$.

- $\text{q3}\downarrow \frac{[A \wp C] \triangleleft B}{\langle A \triangleleft B \rangle \wp C}$: Without loss of generality, we assume that $A = a_n^\perp$ and $B = a_1$ and $C = L_i$ for some $i \in \{2, \dots, n\}$. Then $Q \equiv' \langle [a_n^\perp \wp L_i] \triangleleft a_1 \rangle \wp L_2 \wp \dots \wp L_{i-1} \wp L_{i+1} \wp \dots \wp L_n$, which contains the cycle $\langle a_{i-1}^\perp \triangleleft a_1 \rangle \wp L_2 \wp \dots \wp L_{i-1}$.

- $\text{q3}\downarrow \frac{A \triangleleft [B \wp C]}{\langle A \triangleleft B \rangle \wp C}$: As before, without loss of generality, we assume that $A = a_n^\perp$ and $B = a_1$ and $C = L_i$ for some $i \in \{2, \dots, n\}$. Then $Q \equiv' \langle a_n^\perp \triangleleft [a_1 \wp L_i] \rangle \wp L_2 \wp \dots \wp L_{i-1} \wp L_{i+1} \wp \dots \wp L_n$, which contains the cycle $\langle a_n^\perp \triangleleft a_i \rangle \wp L_{i+1} \wp \dots \wp L_n$.

- $\text{q2}\downarrow \frac{A \triangleleft B}{A \wp B}$: We can assume that $A = L_i$ and $B = L_j$ for some $i, j \in \{1, \dots, n\}$. There are two subcases:

- $i < j$: Then $Q = \langle L_i \triangleleft L_j \rangle \wp L_1 \wp \dots \wp L_{i-1} \wp L_{i+1} \wp \dots \wp L_{j-1} \wp L_{j+1} \wp \dots \wp L_n$ which contains the cycle $L_1 \wp \dots \wp L_{i-1} \wp \langle a_{i-1}^\perp \triangleleft a_j \rangle \wp L_{j+1} \wp \dots \wp L_n$.
- $j < i$: Then $Q = \langle L_i \triangleleft L_j \rangle \wp L_1 \wp \dots \wp L_{j-1} \wp L_{j+1} \wp \dots \wp L_{i-1} \wp L_{i+1} \wp \dots \wp L_n$ which contains the cycle $\langle a_{i-1}^\perp \triangleleft a_j \rangle \wp L_{j+1} \wp \dots \wp L_{i-1}$.

- $\text{s}_3 \frac{[A \wp C] \otimes B}{(A \otimes B) \wp C}$: This case is analogous to the case $\text{q3}\downarrow$ above.

- $\text{s}_2 \frac{A \otimes B}{A \wp B}$: This case is analogous to the case $\text{q2}\downarrow$ above.

In all cases the size of the cycle in Q is strictly smaller than $|Q| = |P|$. \square

Lemma 3.11. *Let P be a balanced formula that contains a cycle. Then P is not provable in BV.*

Proof. Let H be the cycle in P , and let $n = |H|$ be its size. We proceed by induction on n . Note that n has to be even. For $n = 2$, we have that $H \equiv a^\perp \triangleleft a$ or $H \equiv a^\perp \otimes a$ for some atom a . By way of contradiction, assume P is provable in BV. By Lemma 3.6, H is also provable in BV, which is impossible. For the inductive case let now $n > 2$. As before, we have by Lemma 3.6 that H is provable in BV. By Proposition 2.69 and Proposition 2.75, H is provable in BV \hat{u} . Let δ be that proof in BV \hat{u} . Let now Q be the premise of the bottom-most rule instance r of δ that is not a \equiv' (i.e., the conclusion of r is $H' \equiv' H$ and $Q \not\equiv' H$). By Lemma 3.10, Q contains a cycle whose size is smaller than n . By induction hypothesis Q is not provable in BV, and therefore also not provable in BV \hat{u} , which is a contradiction to the existence to δ . \square

We can now complete the proof of Theorem 3.1.

Proof of Theorem 3.1. Let δ be a proof in BV. By Proposition 3.2, there is a balanced formula P , such that $\llbracket P \rrbracket = \llbracket \delta \rrbracket$, and such that P is provable in BV. Now assume, by way of contradiction, that $\llbracket \delta \rrbracket$ is incorrect. That means that $\llbracket \delta \rrbracket$ contains a chordless \ae -cycle, or equivalently, that P contains a cycle. By Lemma 3.11, P is not provable in BV. Contradiction. \square

3.2. Pomset Logic is not Contained in BV. In this section we present a formula that is provable in pomset logic, i.e., has a correct pomset logic proof net, but that is not provable in BV. From what has been said in the previous section, it follows that if such a formula exists then there is also a balanced such formula. The formula we discuss in this section is the formula Q shown below:¹³

$$Q = (\langle a \triangleleft b \rangle \otimes \langle c \triangleleft d \rangle) \wp (\langle e \triangleleft f \rangle \otimes \langle g \triangleleft h \rangle) \wp \langle a^\perp \triangleleft h^\perp \rangle \wp \langle e^\perp \triangleleft b^\perp \rangle \wp \langle g^\perp \triangleleft d^\perp \rangle \wp \langle c^\perp \triangleleft f^\perp \rangle \quad (3.1)$$

or equivalently, the flat sequent

$$\Gamma_Q = [\langle a \triangleleft b \rangle \otimes \langle c \triangleleft d \rangle, \langle e \triangleleft f \rangle \otimes \langle g \triangleleft h \rangle, a^\perp \triangleleft h^\perp, e^\perp \triangleleft b^\perp, g^\perp \triangleleft d^\perp, c^\perp \triangleleft f^\perp] \quad (3.2)$$

Since the formula Q (resp. the sequent Γ_Q) is balanced, there is a unique axiom linking and therefore a unique cographic RB-prenet and a unique tree-like RB-prenet. In Figure 7, we show the tree-like RB prenet for Γ_Q , and on the left of Figure 8 we show the cographic RB-prenet, which is the same for Q and Γ_Q .

To see that these are provable in pomset logic, we have to show that the RB-prenets do not contain chordless \ae -cycles. For this we focus on the tree-like RB-prenet, because in tree-like RB-prenets all \ae -paths (and therefore also all \ae -cycles) are chordless. Hence, it suffices to show that there are no \ae -cycles.

Observe that the B -edges corresponding to the roots of the formulas in Γ_Q cannot participate in an \ae -cycle because they have no adjacent R -edge at the bottom. We can therefore remove each of these B -edges, together with the two adjacent R -edges at the top. The resulting graph is shown on the right of Figure 8.

¹³We will explain in Remark 3.15 how this formula has been found.

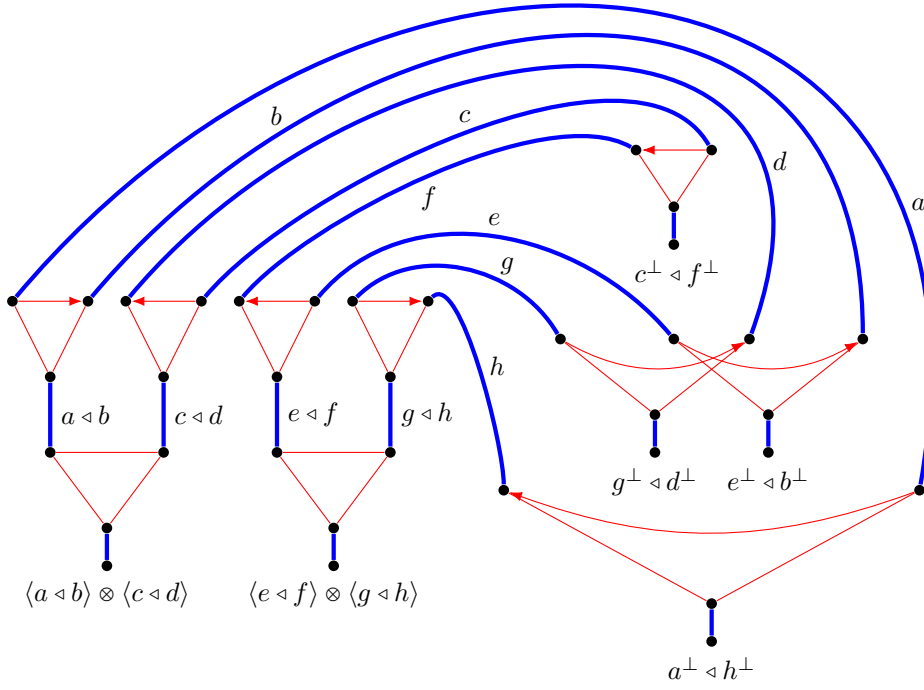


FIGURE 7. The tree-like RB-prenet for the sequent Γ_Q in Equation (3.2)

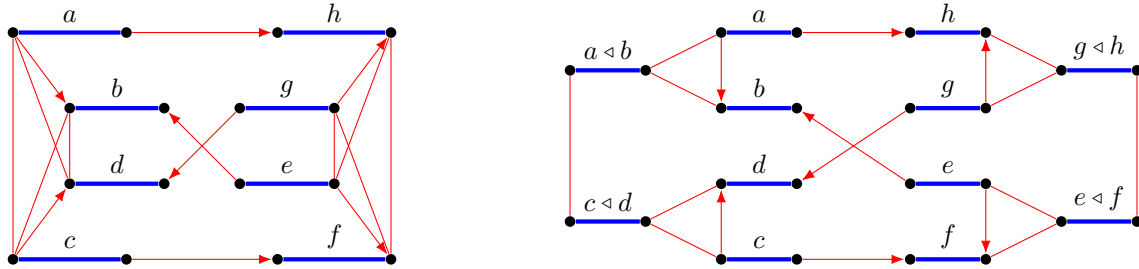


FIGURE 8. Left: The cographic RB-prenet for Q in Equation (3.1) and Γ_Q in Equation (3.2)

Right: A simplification of the tree-like RB-prenet in Figure 7

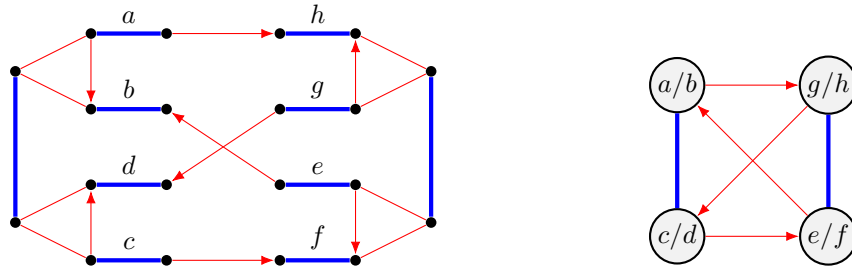


FIGURE 9. Two further simplifications of the graph on the right of Figure 8

Another simplification we can do without affecting the \ae -cycles in the graph is replacing the two B -edges labeled $a \triangleleft b$ and $c \triangleleft d$, together with the connecting R -edge by a single B -edge, and similarly for the two B -edges $g \triangleleft h$ and $e \triangleleft f$. The result is shown on the left of Figure 9.

Finally, observe that there is no \ae -cycle that passes through the two B -edges labeled b and a . The reason is that the directed R -edge between them has the opposite direction of the two adjacent R -edges on the other endpoints of these B -edges. Thus, we can collapse these two edges (and the adjacent “triangle”) to a single vertex. The same can be done for the pairs c/d and g/h and e/f . The result of this operation is shown on the right of Figure 9.

Proposition 3.12. *The formula Q and the sequent Γ_Q shown in Equation (3.1) and Equation (3.2) above are provable in pomset logic.*

Proof. In the paragraphs above, we have argued that the tree-like RB-prenet in Figure 7 has an \ae -cycle if and only if the RB-digraph on the right of Figure 9 has an \ae -cycle. Now it is easy to see that this graph has no \ae -cycle. Hence, tree-like RB-prenet for Γ_Q is correct. \square

Let us now show that the formula Q is not provable in BV. To do so we will show that whenever a BV inference has as conclusion Q then its premise defines an incorrect RB-prenet in pomset logic, and is therefore not provable in pomset logic. Since by Theorem 3.1 all BV proofs induce correct pomset proof nets, we can conclude that those premises are not BV-provable, therefore there is no way to build a BV-proof of Q .

The main difficulty here is to make sure that we do not overlook any case when checking all possible inferences that have Q as conclusion. Since the unit \mathbb{I} can make these kind of arguments difficult to check, we use here $\text{BV}\hat{u}$. Now observe that Q has no subformula of the form $x \otimes x^\perp$. This means we only have to consider the non-axiom rules of $\text{BV}\hat{u}$.

To cut down the number of cases to consider, we take advantage of the symmetries of Q . Let us first look at the *automorphisms*, i.e., permutations of the variables that results in a formula Q' with $Q' \equiv Q$, which means $\llbracket Q' \rrbracket = \llbracket Q \rrbracket$. The following are automorphisms:

- (α) $a \leftrightarrow c, b \leftrightarrow d, e \leftrightarrow g, f \leftrightarrow h$
- (β) $a \mapsto e, b \mapsto f, c \mapsto g, d \mapsto h, e \mapsto c, f \mapsto d, g \mapsto a, h \mapsto b$

The action of these automorphisms on the subformulas of Q of the form $x^\perp \triangleleft y^\perp$ is transitive: $\alpha(a^\perp \triangleleft h^\perp) = c^\perp \triangleleft f^\perp$, $\beta(a^\perp \triangleleft h^\perp) = e^\perp \triangleleft b^\perp$ and $\alpha \circ \beta(a^\perp \triangleleft h^\perp) = g^\perp \triangleleft d^\perp$.

Another useful symmetry is not quite an automorphism: it is the following *anti-automorphism*:

- (γ) $a \leftrightarrow h, b \leftrightarrow g, c \leftrightarrow f, d \leftrightarrow e$

that sends Q to its “conjugate” Q^\dagger defined inductively as follows:

$$x^\dagger = x \text{ when } x \text{ is an atom} \quad (B \odot C)^\dagger = C^\dagger \odot B^\dagger \text{ for } \odot \in \{\otimes, \otimes, \triangleleft\}$$

Note that the reversal of the arguments only matters for the non-commutative connective \triangleleft , and $\llbracket Q^\dagger \rrbracket$ is the same as $\llbracket Q \rrbracket$, except that all directed R -edges have the opposite direction. Thus, conjugacy preserves provability both in pomset logic (reversing the direction of all cycles in the correctness criterion) and in system $\text{BV}\hat{u}$ (the inference rules are closed under conjugacy, with $\hat{q}_3^\perp \downarrow$ and $\hat{q}_3^R \downarrow$ being swapped).

We will now go through all the rules of $\text{BV}\hat{u}$ and check all possible applications. Using a similar argument as in the proof of Lemma 3.10, we will see that in each case there is a cycle in the resulting premise.

- $\mathfrak{q}_4\downarrow \frac{[A \wp C] \triangleleft [B \wp D]}{\langle A \triangleleft B \rangle \wp \langle C \triangleleft D \rangle}$: Because of the action of the automorphisms α/β , we can without loss of generality assume that $A = a^\perp$ and $B = h^\perp$. There are three subcases:
 - $C = e^\perp$ and $D = b^\perp$. We get the cycle $(e \otimes h) \wp \langle e^\perp \triangleleft h^\perp \rangle$ in the premise of the $\mathfrak{q}_4\downarrow$ -application.
 - $C = g^\perp$ and $D = d^\perp$. We get the cycle $(a \otimes d) \wp \langle a^\perp \triangleleft d^\perp \rangle$ in the premise of the $\mathfrak{q}_4\downarrow$ -application.
 - $C = c^\perp$ and $D = f^\perp$. We get the cycle $(b \otimes c) \wp (e \otimes h) \wp \langle c^\perp \triangleleft h^\perp \rangle \wp \langle e^\perp \triangleleft b^\perp \rangle$ in the premise of the $\mathfrak{q}_4\downarrow$ -application.
- $\hat{\mathfrak{q}}_3^1\downarrow \frac{[A \wp C] \triangleleft B}{\langle A \triangleleft B \rangle \wp C}$: As before, because of the symmetries of Q , we only need to consider the case where $A = a^\perp$ and $B = h^\perp$. There are now five subcases of how to match C :
 - $C = \langle a \triangleleft b \rangle \otimes \langle c \triangleleft d \rangle$. We get the cycle $(e \otimes h) \wp \langle b \triangleleft h^\perp \rangle \wp \langle e^\perp \triangleleft b^\perp \rangle$ in the premise of the $\hat{\mathfrak{q}}_3^1\downarrow$ -application.
 - $C = \langle e \triangleleft f \rangle \otimes \langle g \triangleleft h \rangle$. We get the cycle $h \triangleleft h^\perp$ in the premise of the $\hat{\mathfrak{q}}_3^1\downarrow$ -application.
 - $C = e^\perp \triangleleft b^\perp$. We get the cycle $(e \otimes h) \wp \langle e^\perp \triangleleft h^\perp \rangle$ in the premise of the $\hat{\mathfrak{q}}_3^1\downarrow$ -application.
 - $C = g^\perp \triangleleft d^\perp$. We get the cycle $(b \otimes d) \wp (e \otimes h) \wp \langle d^\perp \triangleleft h^\perp \rangle \wp \langle e^\perp \triangleleft b^\perp \rangle$ in the premise of the $\hat{\mathfrak{q}}_3^1\downarrow$ -application.
 - $C = c^\perp \triangleleft f^\perp$. We get the cycle $(f \otimes h) \wp \langle f^\perp \triangleleft h^\perp \rangle$ in the premise of the $\hat{\mathfrak{q}}_3^1\downarrow$ -application.
- $\hat{\mathfrak{q}}_3^R\downarrow \frac{A \triangleleft [B \wp C]}{\langle A \triangleleft B \rangle \wp C}$: Similar to $\hat{\mathfrak{q}}_3^1\downarrow$, by conjugacy.
- $\mathfrak{q}_2\downarrow \frac{A \triangleleft B}{A \wp B}$: The possible values for the ordered pair (A, B) are all pairs of distinct formulas in the sequent Γ_Q in Equation (3.2). We first look at the case $A = \langle a \triangleleft b \rangle \otimes \langle c \triangleleft d \rangle$ and $B = \langle e \triangleleft f \rangle \otimes \langle g \triangleleft h \rangle$. Here we get the cycle $\langle d \triangleleft g \rangle \wp \langle g^\perp \triangleleft d^\perp \rangle$ in the premise. The case $A = \langle e \triangleleft f \rangle \otimes \langle g \triangleleft h \rangle$ and $B = \langle a \triangleleft b \rangle \otimes \langle c \triangleleft d \rangle$ is symmetric to the this one via the automorphism β . Otherwise, either A or B (or both) have the form $x^\perp \triangleleft y^\perp$. It suffices to treat all the cases $R = x^\perp \triangleleft y^\perp$. This is because conjugation exchanges the roles of A and B in the $\mathfrak{q}_2\downarrow$ -rule, and Q is equal to its own conjugate up to the variable renaming performed by γ . We may also without loss of generality assume that $A = a^\perp \triangleleft h^\perp$; as before, this relies on the transitive action of the automorphisms of Q on the $x^\perp \triangleleft y^\perp$ that it contains. There are now five cases for B :
 - $B = \langle a \triangleleft b \rangle \otimes \langle c \triangleleft d \rangle$. We get the cycle $a^\perp \triangleleft a$ in the premise.
 - $B = \langle e \triangleleft f \rangle \otimes \langle g \triangleleft h \rangle$. We get the cycle $h^\perp \triangleleft h$ in the premise.
 - $B = e^\perp \triangleleft b^\perp$. We get the cycle $(e \otimes h) \wp \langle h^\perp \triangleleft e^\perp \rangle$ in the premise.
 - $B = g^\perp \triangleleft d^\perp$. We get the cycle $(a \otimes d) \wp \langle a^\perp \triangleleft d^\perp \rangle$ in the premise.
 - $B = c^\perp \triangleleft f^\perp$. We get the cycle $(f \otimes h) \wp \langle h^\perp \triangleleft f^\perp \rangle$ in the premise.
- $\hat{\mathfrak{s}}_3 \frac{[A \wp C] \otimes B}{(A \otimes B) \wp C}$: There are two possibilities to match $A \otimes B$: either with $\langle a \triangleleft b \rangle \otimes \langle c \triangleleft d \rangle$ or with $\langle e \triangleleft f \rangle \otimes \langle g \triangleleft h \rangle$. Due to the commutativity of \otimes , we have four possibilities to match A and B . Due to the symmetries discussed above, we only need to consider the case where $A = a \triangleleft b$ and $B = c \triangleleft d$. There are now five cases how to match C :
 - $C = \langle e \triangleleft f \rangle \otimes \langle g \triangleleft h \rangle$. We get the cycle $(f \otimes c) \wp \langle c^\perp \triangleleft f^\perp \rangle$ in the premise.
 - $C = a^\perp \triangleleft h^\perp$. We get the cycle $(h^\perp \otimes c) \wp \langle c^\perp \triangleleft f^\perp \rangle \wp (f \otimes h)$ in the premise.
 - $C = e^\perp \triangleleft b^\perp$. We get the cycle $(e^\perp \otimes d) \wp \langle g^\perp \triangleleft d^\perp \rangle \wp (e \otimes g)$ in the premise.

- $C = g^\perp \triangleleft d^\perp$. We get the cycle $d^\perp \otimes d$ in the premise.
- $C = c^\perp \triangleleft f^\perp$. We get the cycle $c^\perp \otimes c$ in the premise.
- $\hat{s}_2 \frac{A \otimes B}{A \wp B}$: This case is already subsumed by the case for $\hat{q}_2 \downarrow$.

In this way, we have completed the proof of the following proposition.

Proposition 3.13. *The formula Q shown in Equation (3.1) is not provable in BV.*

Theorem 3.14. *The theorems of BV form a proper subset of the theorems of pomset logic.*

Proof. This follows immediately from Propositions 3.12 and 3.13. □

Remark 3.15. Let us end this section by some explanation of how the formula Q has been found. Our starting point was the so-called medial rule from system SKS [BT01], a formulation of classical logic in the calculus of structures:

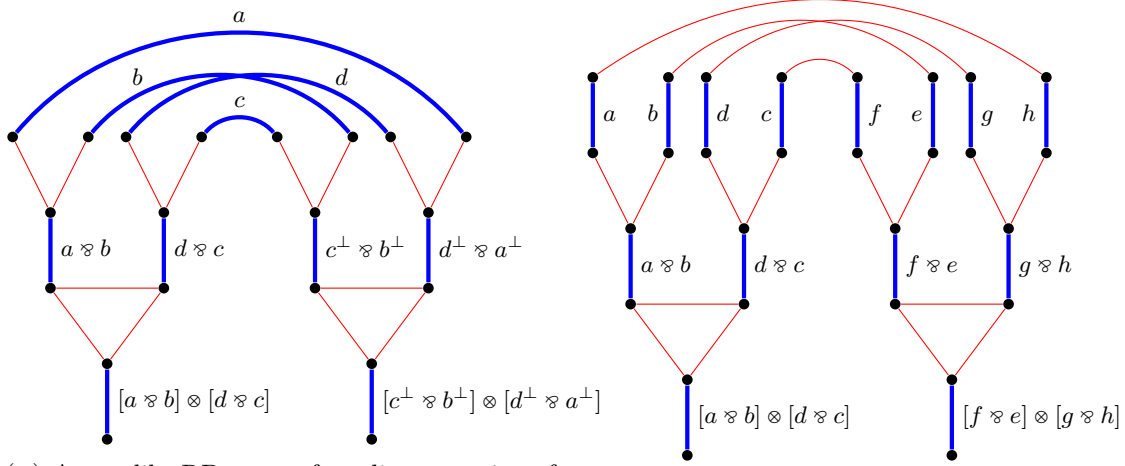
$$\text{m} \frac{(A \wedge D) \vee (B \wedge C)}{[A \vee B] \wedge [D \vee C]} \quad (3.3)$$

which corresponds to the classical tautology $(A \wedge D) \vee (B \wedge C) \Rightarrow [A \vee B] \wedge [D \vee C]$. Its linear version $(A \otimes D) \wp (B \otimes C) \multimap [A \wp B] \otimes [D \wp C]$ is, of course, not a theorem of linear logic. This can be immediately seen by inspecting the RB-prenet for the formula $(a \otimes d) \wp (b \otimes c) \multimap [a \wp b] \otimes [c \wp d]$, which is shown in Figure 10a, and which contains several (chordless) \wp -cycles. Then, on the right of that “medial RB-prenet”, in Figure 10b, we replace the B -edges corresponding to the atoms by a pair of B -edges connected by an (undirected) R -edge. This does not affect provability, as no \wp -cycles are added or removed. Then, in Figure 10c, we give these new R -edges a direction. By choosing the right direction, we can break all \wp -cycles, which means the result becomes correct with respect to the pomset logic correctness criterion. But the resulting formula (or sequent) remains unprovable in BV. To simplify the proof of non-provability in BV, we added further R -edges, as shown in Figure 10d, that do not break provability in pomset logic. It is easy to see that the RB-prenet in Figure 10d is an intermediate step between the one in Figure 7 and the one on the right in Figure 8.

In Example 4.18 we shall see that Figure 10c is also related to a construction that we use for complexity-theoretic purposes; more than that, we shall explain how complexity considerations allowed us to restrict the search space for a formula separating pomset logic from BV.

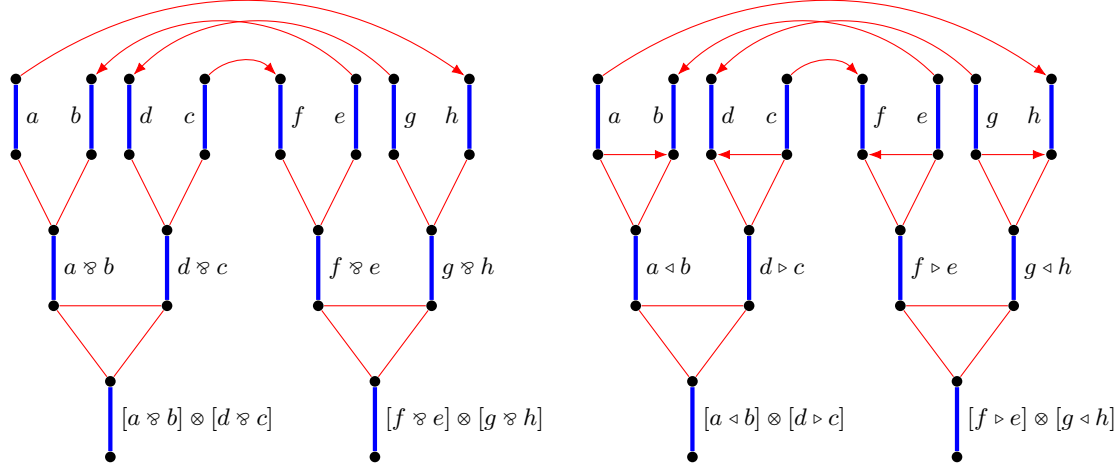
4. COMPLEXITY OF PROVABILITY

After having established that BV and pomset logic are not the same, the next natural question is whether they have the same or different provability complexity. It had already been established before that BV is NP-complete. We recall the proof here and establish a slightly more general result. Then we discuss the complexity of pomset logic and show that already checking the correctness of a pomset logic proofnet is coNP-complete. Based on this observation, we can then show that provability in pomset logic is Σ_2^P -complete.



(A) A tree-like RB-prenet for a linear version of the medial rule of system SKS (cf. Remark 3.15). Note that it does *not* satisfy the MLL+mix correctness criterion, and therefore also not the pomset criterion.

(B) A variation of the prenet on the left. The undirected R -edges on the top correspond to the addition of $a^\perp \otimes h^\perp, b^\perp \otimes e^\perp, d^\perp \otimes g^\perp, c^\perp \otimes f^\perp$. Note that the prenet is still *not* correct.



(C) The R -edges on top are now directed, corresponding to $a^\perp \triangleleft h^\perp, b^\perp \triangleright e^\perp, d^\perp \triangleright g^\perp, c^\perp \triangleleft f^\perp$. This modification validates the pomset logic correctness criterion, but the resulting sequent is not provable in BV.

(D) Adding more R -edges does preserve provability in pomset logic, but showing that the resulting sequent is not provable in BV is easier now, as every possible rule application breaks pomset correctness.

FIGURE 10. From the medial of SKS to our counterexample (cf. Remark 3.15)

4.1. Preliminaries on Complexity Theory and Boolean Formulas. We assume that the reader is familiar with the complexity classes \mathbf{P} , \mathbf{NP} and \mathbf{coNP} (see for instance the reference textbook [AB09, Chapter 2]). \mathbf{NP} and \mathbf{coNP} form the first level of the *polynomial hierarchy* [AB09, Chapter 5], and we shall also be concerned with its *second level* which

contains the classes Σ_2^P and Π_2^P . These are dual in the same way that \mathbf{NP} and \mathbf{coNP} are: a decision problem is in Π_2^P if and only if its negation is in Σ_2^P .

To show that a problem is in Σ_2^P , the most convenient way is perhaps to use the definition in terms of oracle machines.

Definition 4.1 ([AB09, Section 5.5]). Σ_2^P is \mathbf{NP} extended with an \mathbf{NP} oracle; this is usually written as $\Sigma_2^P = \mathbf{NP}^{\mathbf{NP}}$. In other words, a decision problem is in Σ_2^P if and only if it can be solved by an \mathbf{NP} algorithm that can call constant time subroutines for problems in \mathbf{NP} . Similarly, Π_2^P is \mathbf{coNP} extended with an \mathbf{NP} oracle: $\Pi_2^P = \mathbf{coNP}^{\mathbf{NP}}$.

Conversely, to show hardness results, we use complete problems involving *Boolean formulas*.¹⁴ We consider a fixed set of (*Boolean*) *variables*. A *literal* is either x or $\neg x$ for some variable x ; a *clause* is a finite set of literals; a *conjunctive normal form (CNF)* is a finite set of clauses. The idea is that a CNF represents a Boolean formula, as in the following example:

$$\{\{x, y, z\}, \{\neg x, y\}, \{\neg y, \neg z\}\} \rightsquigarrow (x \vee y \vee z) \wedge (\neg x \vee y) \wedge (\neg y \vee \neg z)$$

Consistent with this interpretation, a clause is said to be *satisfied* by some assignment from variables to Booleans in $\{\mathbf{true}, \mathbf{false}\}$ if it contains *some* literal l such that for some variable x , either $l = x$ and x is set to \mathbf{true} , or $l = \neg x$ and x is set to \mathbf{false} ; and an assignment is said to *satisfy* a CNF when it satisfies *all* its clauses. The celebrated *Cook-Levin theorem* states that finding a satisfying assignment for a CNF is \mathbf{NP} -complete. We recall its generalization to the first two levels of the polynomial hierarchy.

Definition 4.2. The problem CNF-SAT consists in deciding, given a CNF as input, whether it admits a satisfying assignment. It is generalized by $\forall\exists$ -CNF-SAT, which takes as input

- a finite set of *universal variables* $X = \{x_1, \dots, x_n\}$,
- a finite set of *existential variables* $Y = \{y_1, \dots, y_m\}$, disjoint from X ,
- and a CNF whose variables are included in $X \cup Y$,

the question being whether *every* partial assignment $X \rightarrow \{\mathbf{true}, \mathbf{false}\}$ can be extended to *some* assignment $X \cup Y \rightarrow \{\mathbf{true}, \mathbf{false}\}$ that satisfies the input CNF.

Theorem 4.3 ([Wra76, Corollary 6]). *CNF-SAT is \mathbf{NP} -complete and $\forall\exists$ -CNF-SAT is Π_2^P -complete.*

As an example, the above-mentioned CNF $\{\{x, y, z\}, \{\neg x, y\}, \{\neg y, \neg z\}\}$, with the universal variables $\{x, y\}$ and the existential variable $\{z\}$, is a negative instance of $\forall\exists$ -CNF-SAT since the corresponding quantified Boolean formula

$$\forall x \forall y \exists z. (x \vee y \vee z) \wedge (\neg x \vee y) \wedge (\neg y \vee \neg z)$$

(where the quantifiers range over $\{\mathbf{true}, \mathbf{false}\}$) is false: for $x = \mathbf{true}$ and $y = \mathbf{false}$, there is no choice of z that satisfies the clause $\neg x \vee y$. Let us conclude these preliminaries by mentioning that \mathbf{coNP} and Σ_2^P admit complete problems involving formulas in *disjunctive normal forms*.

Remark 4.4. There is a standard reduction from CNF-SAT to the case where *all variables occur at least once positively and at least once negatively*, which goes as follows. You can detect in polynomial time for which variables this is not the case, e.g. x has only positive

¹⁴For all the complexity classes that we consider, hardness and completeness are defined with respect to many-to-one polynomial time reductions, as usual; weaker reductions should also work.

occurrences and y has only negative occurrences. Then if you delete all clauses in which either x or $\neg y$ appear (or both), this does not change whether the set of clauses is satisfiable (because if you set $x = \mathbf{true}$ and $y = \mathbf{false}$ you satisfy all deleted clauses and do not lose any degree of freedom for the remaining clauses). After this deletion maybe some other variable occurs with only one polarity, so you need to iterate that procedure. But as the number of iterations is bounded by the number of variables, this reduction is polynomial time.

Thus, this restriction of CNF-SAT is **NP**-complete. For similar reasons, the instances of $\forall\exists$ -CNF-SAT in which no atom appears with a single polarity are also Π_2^P -complete.

4.2. BV is NP-Complete (and how to Generalize Membership in NP). Let us first recall that the complexity of proof search in BV is already known.

Theorem 4.5 (Kahramanođulları [Kah08]). *Provability in system BV is NP-complete.*

The **NP**-hardness part already applies to provability for MLL+mix [Kah08, Corollary 4.5], which BV conservatively extends; we shall not dwell on this here. We will merely remark that while Kahramanođulları proves that MLL+mix is **NP**-hard using a syntactic analysis of the calculus of structures, one could presumably adapt the more traditional *phase semantics* methods used to study the hardness of other variants of linear logic in order to get an alternative proof.

Membership in **NP** is more interesting for us, since it demonstrates a complexity gap with the Σ_2^P -complete pomset logic (unless **NP** = **coNP**), as we already said. We recall here the proof that provability in BVu is in **NP** (the unit-free version is slightly more convenient), giving more details than [Kah08].

The first main argument is a bound on the length of proofs, relying on the following property.

Definition 4.6. We say that a rewriting system \rightsquigarrow on unit-free formulas is *dicograph-monotone* when $\equiv' \subseteq \rightsquigarrow$ and for any $A \rightsquigarrow B$ such that $A \not\equiv' B$:

- either $\frac{B}{A}$ is an instance of an interaction rule of BVu (i.e., one of $\mathbf{ai}^{\circ}\downarrow$, $\mathbf{ai}^{\otimes}\downarrow$, $\mathbf{ai}^{\leftarrow}\downarrow$, $\mathbf{ai}^{\leftarrow}_R\downarrow$);
- or $E_{\llbracket A \rrbracket} \not\prec E_{\llbracket B \rrbracket}$ for a suitable identification of atom occurrences of A and B ensuring that we can consider the vertex sets $V_{\llbracket A \rrbracket}$ and $V_{\llbracket B \rrbracket}$ to be equal.

Proposition 4.7. $A \rightsquigarrow B \iff \frac{B}{A}$ in BVu defines a dicograph-monotone rewriting system.

Proof. This is a direct consequence of the easy direction (“if”) of Theorem 2.77. \square

As discussed in Section 2.7, the rules of SBV are originally derived from a characterization of dicograph inclusion. Dicograph-monotonicity is therefore a fundamental aspect of the design of SBV, BV and their variants. We will discuss another example of this notion in Remark 4.12.

Proposition 4.8. *Let \rightsquigarrow be a dicograph-monotone relation and \rightsquigarrow^* be its transitive closure. Then, for any $A \rightsquigarrow^* B$, there exists a path $A = C_1 \rightsquigarrow \dots \rightsquigarrow C_k = B$ with length $k = O(|A|^2)$.*

Proof. Since \equiv' is transitive by definition, if $C \rightsquigarrow D \rightsquigarrow E$ with $C \equiv' D \equiv' E$, then $C \rightsquigarrow E$ directly (by the assumption $\equiv' \subseteq \rightsquigarrow$). Thus, in the path of minimum length between A and B , at least half of the rewrites $C_i \rightsquigarrow C_{i+1}$ satisfy $C_i \not\equiv' C_{i+1}$. Therefore, up to a factor of

two (that gets absorbed in the $O(\cdot)$ notation), it suffices to bound the number of rewrites with $C_i \not\equiv' C_{i+1}$.

The definition of dicograph-monotonicity then gives us two cases. We claim that in both cases, $V_{\llbracket C_i \rrbracket}^2 \setminus E_{\llbracket C_i \rrbracket} \supseteq V_{\llbracket C_{i+1} \rrbracket}^2 \setminus E_{\llbracket C_{i+1} \rrbracket}$. First, if $V_{\llbracket C_i \rrbracket} = V_{\llbracket C_{i+1} \rrbracket}$ and $E_{\llbracket C_i \rrbracket} \subsetneq E_{\llbracket C_{i+1} \rrbracket}$, then this is immediate. Otherwise, C_i is inferred from C_{i+1} by an interaction rule, and then $\llbracket C_{i+1} \rrbracket$ can be identified with an induced subgraph of $\llbracket C_i \rrbracket$ with two fewer vertices, resulting in a strict inclusion between the complement graphs as we wanted.

Therefore, the natural number $|V_{\llbracket C_i \rrbracket}^2 \setminus E_{\llbracket C_i \rrbracket}|$ strictly decreases at each rewriting step such that $C_i \not\equiv' C_{i+1}$. So, starting from A , the number of such steps is bounded by $|V_{\llbracket A \rrbracket}^2 \setminus E_{\llbracket A \rrbracket}| = O(|A|^2)$. (We see that our definition of dicograph-monotonicity is slightly stronger than what we truly need from the point of view of complexity.) \square

From this proposition, we immediately get:

Corollary 4.9. *Any provable formula A in BVu has a proof of length $O(|A|^2)$ and size $O(|A|^3)$.*

Here ‘‘size’’ refers to the complexity-theoretic notion of the number of bits it takes to write out the proof, hence the additional factor of $|A|$ accounting for the size of each intermediate formula in the derivation. Now that we know that we have short proofs, it remains to show that they can be checked efficiently.

Proposition 4.10. *The validity of a proof in BVu can be checked in polynomial time. In other words, (our presentation of) BVu is a Cook–Reckhow proof system [CR79].*

Proof. It suffices to show that the validity of each inference rule can be verified in polynomial time.

For instances of the \equiv' rule, to check that $A \equiv' B$, one can apply a generic recipe for terms over associative and possibly commutative binary operators: compute hereditarily flattened and possibly sorted representations of A and B in polynomial time, and compare them, as sketched in the introduction of [Bas94] for example.

Let us now consider any other rule r of BVu . There exist two formulas A_r and B_r such that the instances of these rules are precisely the inferences of the form

$$\frac{S\{A_r[a_1 := C_1, \dots, a_k := C_k]\}}{S\{B_r[a_1 := C_1, \dots, a_k := C_k]\}}$$

where $S\{\cdot\}$ is a context, $\{a_1, \dots, a_k\}$ is the set of propositional variables that appear in either A or B (often both), and $[a_1 := C_1, \dots, a_k := C_k]$ denotes a parallel substitution of those variables by formulas. For instance, for $r = \mathbf{q}_2\downarrow$, we may take $k = 2$, $A_{\mathbf{q}_2\downarrow} = a_1 \triangleleft a_2$ and $B_{\mathbf{q}_2\downarrow} = a_1 \otimes a_2$.

Given two formulas A and B , our task is to decide in polynomial time whether one may infer B from A using the rule r . To do so, we first enumerate in polynomial time all the pairs $(S\{\cdot\}, A')$ such that $S\{A'\} = A$ (there are $O(|A|)$ many). For each of these pairs, we match A' against the pattern A_r in linear time; if this succeeds, we get a substitution such that $A_r[a_1 := C_1, \dots, a_k := C_k] = A'$. To conclude, we just have to test the equality $S\{B_r[a_1 := C_1, \dots, a_k := C_k]\} = B$. \square

Remark 4.11. In this proof, we have exploited the fact that the \equiv' rules appear explicitly in our formal proofs. This differs from some traditional presentations of logics in the calculus of structures, which consider that the inference rules work over equivalence classes of formulas

modulo \equiv or \equiv' (those classes are called *structures* in [Gug07]). In such a presentation with \equiv' kept implicit, it is less obvious that proofs are still polynomial-time checkable.

Together, Proposition 4.9 and Proposition 4.10 tell us that BVu has polynomially bounded proofs that can be checked in polynomial time. This entails that provability is in \mathbf{NP} . To extend the result from BVu to BV , note that for any formula A with units, there is a unit-free formula A' such that $A \equiv A'$, which is computable from A in polynomial time, and then $\vdash_{\text{BV}} A \iff \vdash_{\text{BVu}} A'$.

Remark 4.12. Let us revisit Tiu’s result [Tiu06] on the incompleteness of “shallow systems” with respect to BV in the light of our complexity results. The main difference between shallow and deep inference is the lack of contextual closure of the rules in the former, i.e.

$$r \frac{B}{A} \not\Rightarrow r \frac{S\{B\}}{S\{A\}}$$

One interpretation put forth for the results of [Tiu06] is that proof systems that can be translated into shallow systems, such as traditional sequent calculi, fail to capture BV and *pomset logic*¹⁵ because of the lack of deep inference.

We claim that in the case of pomset logic, there is an obstruction unrelated to the shallow vs deep distinction. Indeed, the definition for shallow systems in [Tiu06, §6] also enforces the condition that we called dicograph-monotonicity: observe that the relation $A \prec B$ given in [Tiu06, Definition 6.1] is equivalent to $E_{[A]} \subset E_{[B]}$. (Being able to state this is one reason for talking abstractly about rewriting systems instead of working directly in the calculus of structures.)

Therefore, by Proposition 4.8, any shallow system in this sense has polynomially bounded proofs; and if those proofs are also polynomial-time checkable then provability is in \mathbf{NP} . Together with the result of the next section, this shows that these systems cannot capture pomset logic unless $\mathbf{NP} = \mathbf{coNP}$.

4.3. Correctness of Pomset Logic Proof Nets is \mathbf{coNP} -Complete. As an intermediate step towards our eventual hardness result for provability in pomset logic, we first study the *correctness* problem: given a prenet (either tree-like or cographic), does it satisfy the correctness criterion, that is, is it an actual proof net? This can be seen as the special case of provability for *balanced* formulas, as we remarked before. Let us state our result right away.

Theorem 4.13. *The correctness problem for pomset logic proof nets is \mathbf{coNP} -complete. More precisely, given a sequent Γ and a pre-proof (or linking) ℓ , it is \mathbf{coNP} -complete to check the correctness of its cographic RB -prenet $\rho(\Gamma, \ell)$ or of its tree-like RB -prenet $\tau(\Gamma, \ell)$ (those two conditions being equivalent by Theorem 2.59).*

Proof. Let us show an equivalent reformulation: it is \mathbf{NP} -complete to decide whether a pre-proof is *incorrect*. Membership in \mathbf{NP} is immediate: one can build $\tau(\Gamma, \ell)$, whose æ -cycles provide witnesses for incorrectness by definition, in polynomial time; the size of those cycles is bounded by the number of vertices, and they can be checked in polynomial time. As for the proof of \mathbf{NP} -hardness, it is done in two steps.

¹⁵From [Tiu06, §1]: “We conjecture that [system BV and pomset logic] are actually the same logic. The result on the necessity of deep-inference of BV therefore explains to some extent the difficulty in the sequentialization of Pomset logic.”

- We first show, via a polynomial time reduction, that incorrectness is as hard as finding \mathfrak{a} -cycles in *arbitrary* RB-digraphs (Proposition 4.16 and Theorem 4.17).
- We then prove in the next subsection that the existence of \mathfrak{a} -cycles for general RB-digraphs is **NP**-hard (Theorem 4.19). \square

Remark 4.14. Assuming that $\mathbf{P} \neq \mathbf{NP}$, this refutes [Ret97a, Proposition 5] which claims that a “standard breadth search algorithm” can decide pomset proof net correctness in polynomial time. (The issue with this kind of argument is discussed in [Ngu20, §8.1].) This complexity claim was meant to justify that “the proof net syntax is a sensible syntax by itself” [Ret97a, §3]: it is indeed part of the Cook–Reckhow definition of proof system [CR79], as already mentioned in Proposition 4.10, which says that the verification of BV derivations is in \mathbf{P} .

We are now going to fill the steps outlined above to prove the **NP**-hardness part of Theorem 4.13. The reduction step extends the “proofification” construction from [Ngu20, Section 3.2] that sends *undirected* perfect matchings to MLL+mix pre-proof nets. For the sake of clarity, we present our new reduction, with the same name, as a map from arbitrary RB-digraphs to balanced pomset logic sequents. Since the procedure makes arbitrary choices, the result is defined only up to atom renaming and modulo \equiv , but correctness and provability are indeed invariant for these equivalences.

The following definition (which is illustrated by Figure 11) makes use of the notation $R_{\mathcal{G}}^{\otimes}, R_{\mathcal{G}}^{\triangleleft}, R_{\mathcal{G}}^{\triangleright}, R_{\mathcal{G}}^{\otimes}$ introduced in Definition 2.12.

Definition 4.15. Let $\mathcal{G} = (V_{\mathcal{G}}, R_{\mathcal{G}}, B_{\mathcal{G}})$ be an RB-digraph. The *proofification* of \mathcal{G} , denoted by $\Pi(\mathcal{G})$, is the balanced (flat) sequent (defined only modulo \equiv and variable renaming)

$$[C_{u_1} \otimes C_{v_1}, \dots, C_{u_m} \otimes C_{v_m}, D_{e_1}, \dots, D_{e_k}]$$

where:

- $\{(u_1, v_1), (v_1, u_1), \dots, (u_m, v_m), (v_m, u_m)\} = B_{\mathcal{G}}$ and $\{e_1, \dots, e_k\} = R_{\mathcal{G}}^{\triangleleft}$, both of these being non-repeating enumerations;
- for $u \in V_{\mathcal{G}}$, $C_u = a_{u,w_1} \wp \dots \wp a_{u,w_n}$ where w_1, \dots, w_n is a non-repeating enumeration of the neighborhood of u , such that $\{(u, w_1), \dots, (u, w_n)\} = (\{u\} \times V_{\mathcal{G}}) \cap (R_{\mathcal{G}}^{\otimes} \cup R_{\mathcal{G}}^{\triangleleft} \cup R_{\mathcal{G}}^{\triangleright})$, and
 - for each $\{u, v\} \subseteq V_{\mathcal{G}}$ such that $(u, v) \in R_{\mathcal{G}}^{\otimes}$ (equivalently, $(v, u) \in R_{\mathcal{G}}^{\otimes}$), we give the names $a_{u,v}$ and $a_{v,u}$ to a fresh pair of *dual* atoms;
 - for each $(u, v) \in R_{\mathcal{G}}^{\triangleleft}$ (equivalently, $(v, u) \in R_{\mathcal{G}}^{\triangleright}$), we generate two fresh atoms $a_{u,v}$ and $a_{v,u}$ that are *not* dual;
- for $(u, v) \in R_{\mathcal{G}}^{\triangleleft}$, we let $D_{(u,v)} = a_{u,v}^{\perp} \triangleleft a_{v,u}^{\perp}$, using the above-defined atoms.

The ideas that were already present in [Ngu20] are the use of *par* (\wp) to represent vertices, *tensor* (\otimes) for matching edges, and dual atoms for non-matching edges. The novelty here is how the *seq* (\triangleleft) connective of pomset logic serves to encode edge directions through the formulas $D_{(u,v)}$.

Proposition 4.16. *Proofifications can be computed in polynomial time.*

Proof. Immediate from the definition. \square

Since a balanced sequent (or formula) uniquely defines a (tree-like or cographic) prenet, we can define the *correctness* of such a sequent (or formula), to be the correctness of the corresponding prenet.

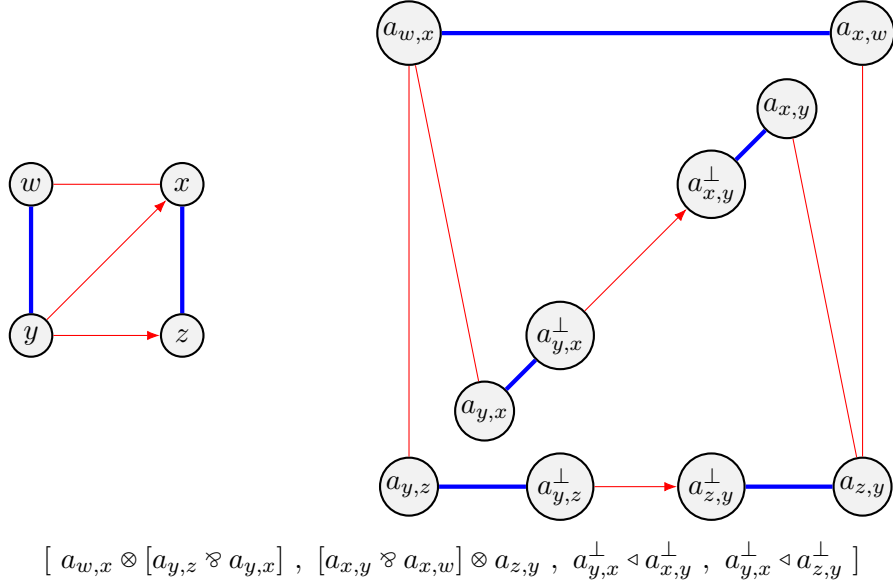


FIGURE 11. An RB-digraph (left) and its proofification (below) together with the corresponding cographic RB-prenet (right). Note that $a_{w,x}$ and $a_{x,w}$ are defined to be dual atoms.

Theorem 4.17. *Let \mathcal{G} be an RB-digraph and $\Pi(\mathcal{G})$ be its proofification. Then \mathcal{G} admits an \mathfrak{a} -cycle if and only if $\Pi(\mathcal{G})$ is incorrect.*

Proof. By slightly adapting and then extending the reasoning found in [Ngu20, Proposition 3.9], one could show that the \mathfrak{a} -cycles in \mathcal{G} are in bijection with those in the tree-like RB-prenet of $\Pi(\mathcal{G})$, and the theorem statement would then follow immediately. However, we find it more convenient here to work with the cographic RB-prenet $\llbracket \Pi(\mathcal{G}) \rrbracket = \rho(\Pi(\mathcal{G}), \ell(\Pi(\mathcal{G})))$ (see Figure 11 for an example).

First, by unfolding the definitions, one can write the set of non-matching edges of $\llbracket \Pi(\mathcal{G}) \rrbracket$ as the disjoint union $R_1 \cup R_2$ where

$$R_1 = \{(a_{u,v}, a_{x,y}) \mid (u,x) \in B_{\mathcal{G}}, (v,y) \in V_{\mathcal{G}}^2\} \quad R_2 = \{(a_{u,v}^\perp, a_{v,u}^\perp) \mid (u,v) \in R_{\mathcal{G}} \wedge (v,u) \notin R_{\mathcal{G}}\}$$

Next, suppose that there exists an \mathfrak{a} -cycle $u_0, \dots, u_{n-1}, u_n = u_0$ in \mathcal{G} , assuming without loss of generality that $(u_i, u_{i+1}) \in B_{\mathcal{G}}$ when i is even and $(u_i, u_{i+1}) \in R_{\mathcal{G}}$ when i is odd. Let us write $a[u, v] = a_{u,v}$ for readability. The aforementioned \mathfrak{a} -cycle can be turned into the cycle

$$a[u_0, u_{n-1}] \rightarrow a[u_1, u_2] \rightarrow^* a[u_2, u_1] \rightarrow \cdots \rightarrow a[u_{n-1}, u_0] \rightarrow^* a[u_0, u_{n-1}]$$

in $\llbracket \Pi(\mathcal{G}) \rrbracket$, where \rightarrow^* denotes a non-matching edge and \rightarrow denotes either a matching edge or an \mathfrak{a} -path of length 3 starting and ending with matching edges. The fact that it is an \mathfrak{a} -cycle is immediate. To show that $(\Pi(\mathcal{G}), \ell(\Pi(\mathcal{G})))$ is incorrect, we must also check that it is chordless, that is, we must rule out the possibility that any edge in $R_1 \cup R_2$ is a chord.

- Let $e = (a_{u,v}, a_{x,y}) \in R_1$, which means that $(u,x) \in B_{\mathcal{G}}$. Suppose that $a_{u,v}$ is part of the cycle in $\llbracket \Pi(\mathcal{G}) \rrbracket$ (otherwise, e is not a chord). Then u is part of the original cycle in \mathcal{G} , i.e. $u = u_i$ for some $i \in \{0, \dots, n-1\}$. Since the cycle in \mathcal{G} is alternating, $x = u_{i-1}$ or $x = u_{i+1}$ (with indexing modulo n) depending on the parity of i , and since it is elementary, this

is the only time x is visited. So there exists a unique $z \in V_{\mathcal{G}}$ such that $a_{x,z}$ belongs to the cycle in $\llbracket \Pi(\mathcal{G}) \rrbracket$. We then have either $(a_{x,z}, a_{u,v}) \in R_1$ or $(a_{u,v}, a_{x,z}) \in R_1$. A case analysis depending on whether $z = y$ then shows that e cannot be a chord.

- The endpoints of edges in R_2 are not incident to any other non-matching edges, so R_2 cannot provide any chord.

Conversely, one can show that any \ae -cycle in $\llbracket \Pi(\mathcal{G}) \rrbracket$ induces in a canonical way an alternating cycle in \mathcal{G} , and if one starts from a chordless \ae -cycle, then the resulting cycle is elementary. \square

Before moving to the last missing ingredient for the proof of Theorem 4.13, let us draw a connection with some previous material in this paper.

Example 4.18. Consider the digraph with perfect matching on the right of Figure 9. Since it has no \ae -cycle, its proofification below is a provable formula in pomset logic.

$$[[a \otimes b] \otimes [c \otimes d], [e \otimes f] \otimes [g \otimes h], a^\perp \triangleleft h^\perp, e^\perp \triangleleft b^\perp, c^\perp \triangleleft f^\perp, g^\perp \triangleleft d^\perp]$$

This is very close to the counterexample presented in Section 3.2. In fact, we have already seen an abridged drawing for the corresponding proof net in that section, in Figure 10c. In general, our complexity results imply that unless $\mathbf{NP} = \mathbf{coNP}$, there must exist some proofification that is provable in pomset logic but not in BV, and this provides an explicit example. Furthermore, if we forget the edge directions in this graph (right of Figure 9) and then take its proofification, we get the “linear medial” of Remark 3.15 / Figure 10a.

4.4. Finding Alternating Elementary Cycles is NP-Hard. The technical heart of our proof of \mathbf{coNP} -hardness for pomset proof net correctness is:

Theorem 4.19. *Deciding the existence of \ae -cycles in RB-digraphs is NP-hard.*

We shall prove this by a many-one polynomial time reduction from the CNF-SAT problem described in Section 4.1. A more concise proof would have been possible by relying on a result on *edge-colored digraphs* [GLMM13, Theorem 5] (this is actually how we discovered the theorem, and this proof can be found in [Ngu19]). Nevertheless, our direct reduction, which is heavily inspired by¹⁶ [GLMM13], has two advantages: it shortens the chain of dependencies, and lends itself to being adapted into the Σ_2^P -hardness proof¹⁷ of our next subsection.

¹⁶One step in the proof of [GLMM13, Theorem 5] consists of adding edge directions to a reduction from digraphs to undirected 2-edge-colored graphs (called Häggkvist’s transformation); morally, our reduction replaces this with a well-known correspondence between digraphs and undirected RB-graphs (something of this kind is implicit, for instance, in the use of the classical Ford-Fulkerson maximum flow algorithm to compute a maximum matching).

¹⁷This is why we use a reduction from CNF-SAT, whereas the proof of [GLMM13, Theorem 5] consists of a reduction between two graph-theoretic problems—no Boolean formulas are mentioned anywhere in [GLMM13].

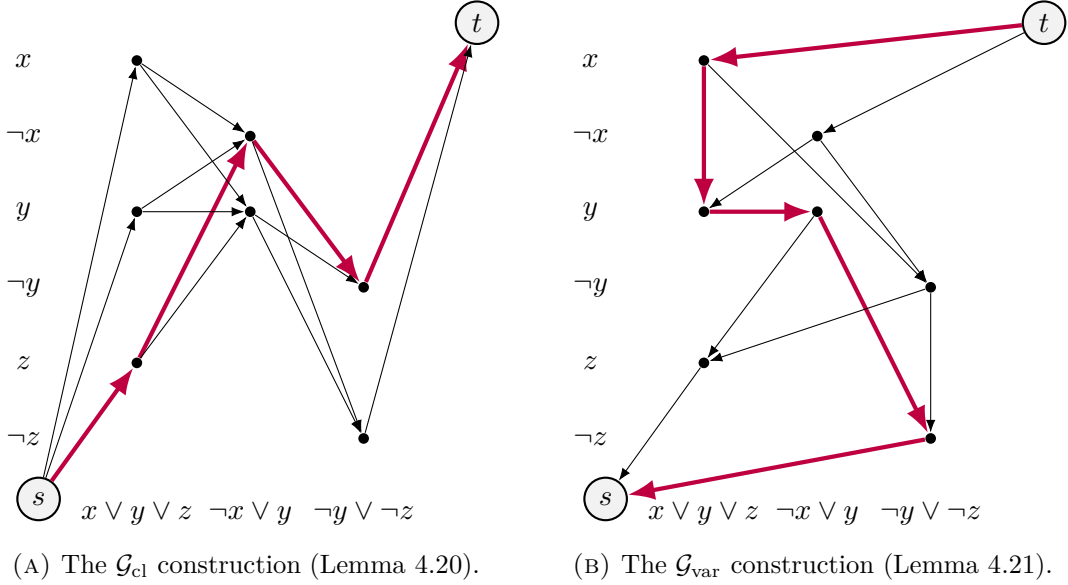


FIGURE 12. The reduction for the proof of Theorem 4.19 on the CNF $(x \vee y \vee z) \wedge (\neg x \vee y) \wedge (\neg y \vee \neg z)$. The thick colored paths correspond: in \mathcal{G}_{cl} (left), to selecting the literals $z, \neg x, \neg y$ in the successive clauses; in \mathcal{G}_{var} (right), to the assignment $x = \text{false}, y = \text{false}, z = \text{true}$ which makes the selected literals true. Observe that the two paths do not share any intermediate vertex. It is also worth paying attention to the horizontal edge for y on the right, that showcases how to handle multiple occurrences of a literal set to false.

In a nutshell. Let us first give a rough idea of the proof, illustrated by figures on the CNF-SAT instance $(x \vee y \vee z) \wedge (\neg x \vee y) \wedge (\neg y \vee \neg z)$.

We first build a digraph \mathcal{G}_{cl} (Figure 12a) with two distinguished vertices s and t such that paths from s to t are in bijection with *choices of one literal per clause*. To make this work, the graph contains one vertex for each literal occurrence.

Next, we build *on the same set of vertices* a digraph \mathcal{G}_{var} (Figure 12b) such that paths from t to s correspond bijectively to variable assignments in the following way: the path traverses all the literal occurrences set to **false**. The point is that if we manage to go from s to t with a path in \mathcal{G}_{cl} , and then go back from t to s with a path in \mathcal{G}_{var} *avoiding all vertices that were already visited*, then the first path will only select literals set to **true** by the second path. This means that cycles visiting both s and t yield *satisfying assignments* and vice versa.

Finally, the tricky part is to reduce finding such a cycle with two prescribed vertices to finding an \ae -cycle in an RB-digraph. This is done by a generic construction (Figure 13) that morally “superimposes” in some way these two graphs \mathcal{G}_{cl} and \mathcal{G}_{var} . It requires a few additional conditions, among which the acyclicity of both \mathcal{G}_{cl} and \mathcal{G}_{var} .

Proof details. For the remainder of this subsection, we fix an instance of CNF-SAT, presented formally as in Section 4.1: it is a finite ordered set of *clauses* $\{C_1, \dots, C_n\}$; each clause is a finite ordered set of *literals* $C_i = \{l_{i,1}, \dots, l_{i,m(i)}\}$; finally, each literal is either

x or $\neg x$ for some variable $x \in X = \{x_1, \dots, x_p\}$. Given this instance, we consider a set of vertices $V_{\text{occ}} = \{v_{i,j} \mid (i,j) \in I\}$ with one vertex for each literal occurrence (thus, $I = \{(i,j) \in \mathbb{N}^2 \mid 1 \leq i \leq n, 1 \leq j \leq m(i)\}$), plus two auxiliary vertices s and t (outside the set V_{occ}).

Recall that a digraph is said to be *acyclic* when it contains no cycles. An observation to keep in mind for the three following lemmas is that a path in an acyclic digraph is always elementary.

Lemma 4.20 (see Figure 12a). *From the given CNF-SAT instance, one can build in polynomial time a directed graph $\mathcal{G}_{\text{cl}} = (V_{\text{occ}} \cup \{s, t\}, E_{\text{cl}})$ such that:*

- \mathcal{G}_{cl} is acyclic, s has no incoming edges and t has no outgoing edges;
- each path from s to t in \mathcal{G}_{cl} visits exactly the intermediate vertices $\{v_{i,j[i]} \mid 1 \leq i \leq n\}$ for some $j[1], \dots, j[n]$ with $1 \leq j[i] \leq m(i)$;
- conversely, every such choice of one literal per clause induces a unique path from s to t , visiting exactly the corresponding vertices.

Proof. As illustrated in Figure 12a, we take:

$$\begin{aligned} E_{\text{cl}} = & \{(s, v_{1,j}) \mid j \in \{1, \dots, m(1)\}\} \\ & \cup \{(v_{i,j}, v_{i+1,j'}) \mid i \in \{1, \dots, n-1\}, j \in \{1, \dots, m(i)\}, j' \in \{1, \dots, m(i+1)\}\} \\ & \cup \{(v_{n,j}, t) \mid j \in \{1, \dots, m(n)\}\} \end{aligned}$$

It is straightforward to check that the required properties hold. For instance, the absence of cycles in \mathcal{G}_{cl} is a consequence of the following fact: for all $(v_{i,j}, v_{i',j'}) \in E_{\text{cl}}$ we have $i < i'$. \square

Lemma 4.21 (see Figure 12b). *From the given CNF-SAT instance, assuming without loss of generality that each variable has at least one positive and one negative occurrence (see Remark 4.4), one can build in polynomial time a directed graph $\mathcal{G}_{\text{var}} = (V_{\text{occ}} \cup \{s, t\}, E_{\text{var}})$ such that:*

- \mathcal{G}_{var} is acyclic, t has no incoming edges and s has no outgoing edges (note that the roles of t and s are reversed compared to \mathcal{G}_{cl});
- for each path from t to s , the set of intermediate vertices in \mathcal{G}_{var} that it visits is of the form $\{v_{i,j} \mid (i,j) \in I, l_{i,j} \in \{\neg x \mid x \in Y\} \cup (X \setminus Y)\}$ for a unique set of variables $Y \subseteq X$;
- conversely, every such subset of variables corresponds to a (unique) path from t to s .

In the last two items above, one should see such a $Y \subseteq X$ as an assignment $\chi_Y : X \rightarrow \{\mathbf{true}, \mathbf{false}\}$, with $Y = \chi_Y^{-1}(\{\mathbf{true}\})$. So the vertices traversed correspond to the literals set to false.

Proof. Let us first describe what the paths starting from t will look like once we have defined the digraph. First, we have to choose $l_1 \in \{x_1, \neg x_1\}$ and go to its first occurrence (first for the order induced by the clauses). Then as long as we are on an occurrence of l_1 which is not the last one, there is a single outgoing edge, and it leads to the next occurrence. Finally, once the last occurrence of l_1 is reached, we may go to the first occurrence of l_2 for some choice $l_2 \in \{x_2, \neg x_2\}$. And so on, until the last occurrence of either x_p or $\neg x_p$ which finally allows us to arrive at s .

To enforce this, we define E_{var} (as shown in Figure 12b) to consist of all the edges:

- $(v_{i,j}, v_{i',j'})$ such that $l_{i,j} = l_{i',j'} = l$ and the occurrence of l in $C_{i'}$ is the successor of its occurrence in C_i , i.e. $i < i'$ and $i < i'' < i' \implies l \notin C_{i''}$;

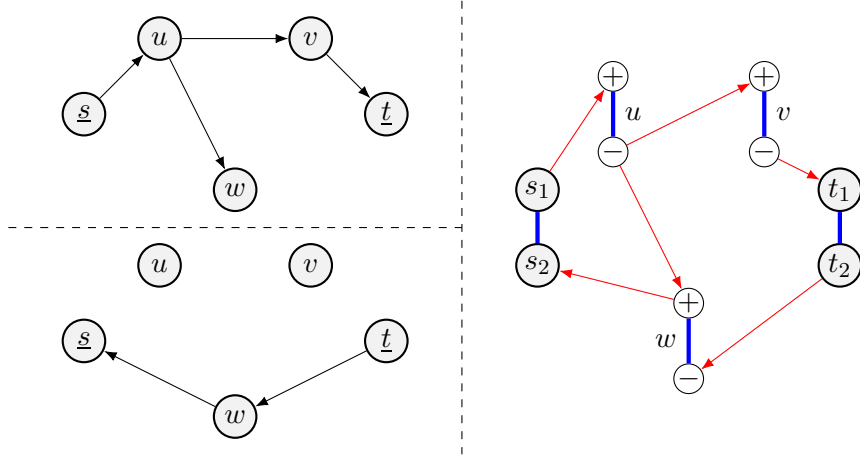


FIGURE 13. Example for the proof of Lemma 4.22. On the left, two digraphs are drawn on the same vertices. On the right, the RB-digraph obtained by applying the construction of Lemma 4.22 with the distinguished vertices s and t . (We do not reuse the graphs in Figure 12 because the drawing of their “superposition” would be unreadable.) The pair of paths $(s \rightarrow u \rightarrow v \rightarrow t, t \rightarrow w \rightarrow s)$ on the left corresponds to an æ-cycle on the right, whereas the “short-circuit” $(s \rightarrow u \rightarrow w, w \rightarrow s)$ – which we want to exclude – corresponds to a cycle that is not alternating (the consecutive edges $u^\ominus \rightarrow w^\oplus \rightarrow s_2$ are both outside the matching).

- $(v_{i,j}, v_{i',j'})$ such that for some $(l_{i,j}, l_{i',j'}) \in \bigcup_{1 \leq k \leq p-1} (\{x_k, \neg x_k\} \times \{x_{k+1}, \neg x_{k+1}\})$, C_i is the last clause containing a literal equal to $l_{i,j}$ while $C_{i'}$ is the first clause containing $l_{i',j'}$;
- $(t, v_{i,j})$ and $(t, v_{i',j'})$, where $l_{i,j} = x_1$, $l_{i',j'} = \neg x_1$ and $C_i, C_{i'}$ are the first clauses in which those literals appear respectively;
- $(v_{i,j}, s)$ and $(v_{i',j'}, s)$ for the last occurrences $l_{i,j}, l_{i',j'}$ of $x_p, \neg x_p$. □

Lemma 4.22 (see Figure 13). *Let $\mathcal{G}_1 = (V, E_1)$ and $\mathcal{G}_2 = (V, E_2)$ be two directed graphs with the same vertex set V . Let $s, t \in V$. Assume that \mathcal{G}_1 and \mathcal{G}_2 are acyclic and that s (resp. t) has no incoming edge in \mathcal{G}_1 (resp. \mathcal{G}_2) and no outgoing edge in \mathcal{G}_2 (resp. \mathcal{G}_1). Then one can build in polynomial time an RB-digraph \mathcal{H} whose æ-cycles are in bijection with the pairs (P_1, P_2) where:*

- P_1 is a path from s to t in \mathcal{G}_1 ;
- P_2 is a path from t to s in \mathcal{G}_2 ;
- $P_1 \setminus \{s, t\}$ and $P_2 \setminus \{s, t\}$ are vertex-disjoint.

Proof. Our construction for $\mathcal{H} = (V_{\mathcal{H}}, R_{\mathcal{H}}, B_{\mathcal{H}})$ associates to each original vertex a matching edge:

$$V_{\mathcal{H}} = \{s_1, s_2, t_1, t_2\} \cup \{v^p \mid v \in V, p \in \{\oplus, \ominus\}\} \quad \text{where } V' = V \setminus \{s, t\}$$

$$B_{\mathcal{H}} = \{(s_1, s_2), (s_2, s_1), (t_1, t_2), (t_2, t_1)\} \cup \{(v^p, v^q) \mid v \in V, (p, q) \in \{(\oplus, \ominus), (\ominus, \oplus)\}\}$$

The non-matching edges are obtained from the original edges: writing E_i for the set of edges of \mathcal{G}_i (for $i \in \{1, 2\}$), we take $R_{\mathcal{H}} = R'_1 \cup R'_2$ where

$$\begin{aligned} R'_1 &= \{(s_1, v^\oplus) \mid v \in V' \text{ such that } (s, v) \in E_1\} \cup \{(u^\ominus, t_1) \mid u \in V' \text{ such that } (u, t) \in E_1\} \\ &\quad \cup \{(u^\ominus, v^\oplus) \mid u, v \in V' \text{ such that } (u, v) \in E_1\} \\ R'_2 &= \{(t_2, v^\ominus) \mid v \in V' \text{ such that } (t, v) \in E_2\} \cup \{(u^\oplus, s_2) \mid u \in V' \text{ such that } (u, t) \in E_2\} \\ &\quad \cup \{(u^\oplus, v^\ominus) \mid u, v \in V' \text{ such that } (u, v) \in E_2\} \end{aligned}$$

so that B , R'_1 and R'_2 are pairwise disjoint.

Given a pair of paths (P_1, P_2) with $P_1 = s \rightarrow u_1 \rightarrow \dots \rightarrow u_r \rightarrow t$ and $P_2 = t \rightarrow v_1 \rightarrow \dots \rightarrow v_q \rightarrow s$, as specified in the lemma statement, we can build an \mathfrak{a} -cycle in \mathcal{H} , where ‘ \Rightarrow ’ denotes a matching edge:

$$s_1 \rightarrow u_1^\oplus \Rightarrow u_1^\ominus \rightarrow u_2^\oplus \Rightarrow \dots \Rightarrow u_r^\ominus \rightarrow t_1 \Rightarrow t_2 \rightarrow v_1^\ominus \Rightarrow v_1^\oplus \rightarrow v_2^\ominus \Rightarrow \dots \Rightarrow v_q^\oplus \rightarrow s_2 \Rightarrow s_1$$

It is alternating by construction, and it is elementary because P_1 and P_2 themselves are necessarily elementary (they are paths in acyclic digraphs).

Conversely, we want to extract such a pair of paths (P_1, P_2) from any \mathfrak{a} -cycle in \mathcal{H} . First, we claim that the RB-digraph $(V_{\mathcal{H}}, R'_1, B)$ does not contain any \mathfrak{a} -cycle. This is because any \mathfrak{a} -path of length ≥ 2 in it is strictly increasing for the following order: the lexicographic product of the transitive closure of E_1 — which, by acyclicity assumption on \mathcal{G}_1 , is a partial order — with the order on $\{\oplus, \ominus\}$ defined by $\oplus \leq \ominus$. Likewise, $(V_{\mathcal{H}}, R'_2, B)$ does not admit any \mathfrak{a} -cycle either. Therefore, an \mathfrak{a} -cycle in \mathcal{H} must contain two edges $e_1 \in R'_1$ and $e_2 \in R'_2$. Given such an \mathfrak{a} -cycle, let π_i be the directed subpath starting with e_i and ending with e_{3-i} . Then:

- π_1 contains a subpath $v_1 \xrightarrow{R'_1} v_2 \xrightarrow{B} v_3 \xrightarrow{R'_2} v_4$. Since v_2 is the target of an edge in E_1 , either $v_2 = t_2$ or $v_2 = v^\oplus$ for some $v \in V$. In the latter case, we have $v_3 = v^\ominus$, which is impossible for the source of an edge in E_2 . Therefore $(v_2, v_3) = (t_1, t_2)$.
- π_2 contains a subpath $u_1 \xrightarrow{R'_2} u_2 \xrightarrow{B} u_3 \xrightarrow{R'_1} u_4$. Similarly to the previous case, we conclude that $(u_2, u_3) = (s_2, s_1)$.

To recapitulate: the cycle must switch at some point from edges in R'_1 to edges in R'_2 , and it must also switch back at some point; it can only do the former by crossing $(t_1, t_2) \in B$ and the latter by crossing $(s_2, s_1) \in B$. Therefore, this \mathfrak{a} -cycle decomposes into an \mathfrak{a} -path P_1 from s_1 to t_1 in $(V_{\mathcal{H}}, R'_1, B)$ and an \mathfrak{a} -path P_2 from t_2 to s_2 in $(V_{\mathcal{H}}, R'_2, B)$, glued together by $(t_1, t_2) \in B$ and $(s_2, s_1) \in B$. These paths are vertex-disjoint because the \mathfrak{a} -cycle that they form is, by definition of \mathfrak{a} -cycle, elementary; they can be lifted to yield the desired pair of paths in \mathcal{G}_1 and \mathcal{G}_2 . \square

We can now combine these ingredients to reduce CNF-SAT to the directed \mathfrak{a} -cycle problem.

Proof of Theorem 4.19. We apply the construction of the previous lemma to \mathcal{G}_{cl} and \mathcal{G}_{var} (with $V = V_{\text{occ}}$). An \mathfrak{a} -cycle in the resulting RB-digraph corresponds to a path P_{cl} from s to t in \mathcal{G}_{cl} plus a path P_{var} from t to s in \mathcal{G}_{var} , that are vertex-disjoint except at s and t . We have to show that the existence of the latter is equivalent to that of an assignment that *satisfies all the clauses*.

Suppose that we are given such an assignment. First, there exists a unique path P_{var} in \mathcal{G}_{var} that visits all literal occurrences set to **false** (Lemma 4.21). Since the assignment

is satisfying, we may choose in each clause a literal set to **true**. This corresponds by Lemma 4.20 to a path P_{cl} in \mathcal{G}_{cl} . If some vertex of V_{occ} were to appear in both P_{var} and P_{cl} , it would mean that the corresponding literal is set both to **false** and to **true** simultaneously.

The converse direction proceeds by a similar reasoning. \square

4.5. Pomset Logic Provability is Σ_2^P -Complete. We are now in a position to treat our main complexity result:

Theorem 4.23. *The provability problem of pomset logic is Σ_2^P -complete.*

As in the case of correctness, we start with the membership part whose proof is easier than the hardness part.

Proposition 4.24. *Pomset logic provability is in Σ_2^P .*

Proof. The size of any pre-proof (i.e. axiom linking) is bounded by a polynomial in the size of its conclusion, and the correctness criterion is in **coNP**. Therefore, there is a $\Sigma_2^P = \mathbf{NP}^{\mathbf{NP}}$ algorithm that consists in first guessing non-deterministically a pre-proof whose conclusion is the input formula, then calling a **NP** oracle for incorrectness on this guess, and finally accepting whenever the oracle returns **false**. \square

To show the Σ_2^P -hardness of provability, we proceed analogously to the **coNP**-hardness proof for correctness, using a sequence of reductions

Boolean formula problem \longrightarrow graph-theoretic problem \longrightarrow pomset logic problem

Here, this pattern is instantiated with the Π_2^P -complete $\forall\exists$ -CNF-SAT problem (cf. Section 4.1) to the left and *unprovability* in pomset logic to the right. For the auxiliary step in the middle, we use a problem formulated using “switchings” of “paired graphs”, as in the standard Danos–Regnier correctness criterion for **MLL+mix** [DR89, FR94].

Definition 4.25. A *paired digraph* is a directed graph \mathcal{G} equipped with a set \mathfrak{P} of unordered pairs of edges, such that the pairs are disjoint (if $p, p' \in \mathfrak{P}$, then $p \cap p' = \emptyset$) and paired edges have the same source (if $\{(u_1, v_1), (u_2, v_2)\} \in \mathfrak{P}$, then $u_1 = u_2$). An edge is *unpaired* if it is not an element of any pair in \mathfrak{P} . A *switching*¹⁸ of $(\mathcal{G}, \mathfrak{P})$ is a spanning subgraph of \mathcal{G} that contains all unpaired edges and exactly one edge in each pair.

A *paired RB-digraph* is a tuple $(V_{\mathcal{G}}, R_{\mathcal{G}}, B_{\mathcal{G}}, \mathfrak{P})$ such that $(V_{\mathcal{G}}, R_{\mathcal{G}}, B_{\mathcal{G}})$ is a RB-digraph and $(V_{\mathcal{G}}, R_{\mathcal{G}}, \mathfrak{P})$ is a paired digraph. Consistently with the above, a *switching* of $(V_{\mathcal{G}}, R_{\mathcal{G}}, B_{\mathcal{G}}, \mathfrak{P})$ is an RB-digraph of the form $(V_{\mathcal{G}}, R', B_{\mathcal{G}}, \mathfrak{P})$ where $R' \subseteq R_{\mathcal{G}}$ contains all unpaired edges from $R_{\mathcal{G}}$ and exactly one edge in each pair.

The reader familiar with proof nets might find it strange that we are mixing perfect matchings and paired graphs together: the former already play the role of expressing the correctness criterion, so what are the latter good for? The answer is that switchings will morally correspond to choices of axiom linkings: the possible proof nets whose conclusion is a given arbitrary (not necessarily balanced) formula only differ by which pairs of dual atoms are joined by axiom links. Without further ado, let us present our reductions.

¹⁸The literature sometimes uses “switching” for a choice function on the set of edge pairs, rather than the digraph defined by this choice.

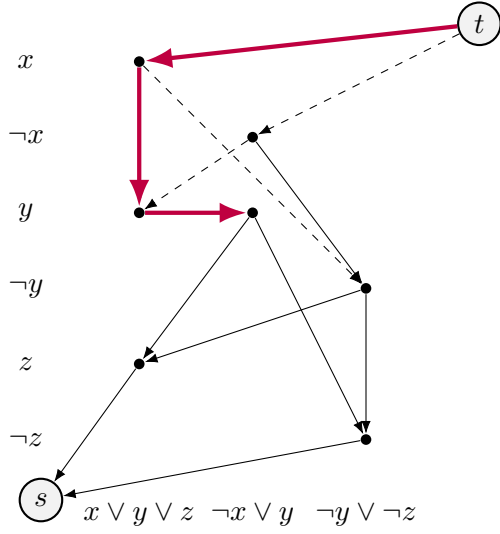


FIGURE 14. The paired \mathcal{G}_{var} construction from the proof of Lemma 4.26, on the $\forall\exists$ -CNF-SAT instance $\forall x \forall y \exists z (x \vee y \vee z) \wedge (\neg x \vee y) \wedge (\neg y \vee \neg z)$ (compare with Figure 12b). The dashed edges represent the choice of a switching in which these edges are erased. The specific choice made here gives us three bits of information that can be interpreted as $x = \text{false}$, $(x = \text{true}) \implies (y = \text{true})$ and $(x = \text{false}) \implies (y = \text{false})$ – recall that visiting a literal corresponds to setting it to **false** – which entails $x = y = \text{false}$. The thick colored path materializes this assignment of the universal variables, and it is a prefix of all paths from t to s in this switching.

Lemma 4.26. *It is Π_2^P -hard to decide whether every switching of a paired RB-digraph contains at least one α -cycle.*

Proof. We proceed by polynomial time reduction from $\forall\exists$ -CNF-SAT (cf. Section 4.1). Consider an instance of this problem whose universal variables are x_1, \dots, x_m (we will not need to directly manipulate the CNF or the existential variables). We reuse the constructions \mathcal{G}_{cl} and \mathcal{G}_{var} of Lemmas 4.20 and 4.21, applying them to the CNF part of the input. Strictly speaking, the digraph \mathcal{G}_{var} depends on the order of variables, and here we shall consider for convenience that the universal variables come before the existential ones, with the i -th variable being x_i for $i \leq m$.

The key idea is to distinguish a set \mathfrak{P} of disjoint edge pairs in \mathcal{G}_{var} (see Figure 14 for an example). Let t and s be its distinguished source and sink vertices. For $i \in \{1, \dots, m\}$, we write:

- u_i (resp. v_i) for the vertex that corresponds to the first (resp. last) occurrence of x_i ;
- u_i^- (resp. v_i^-) for the vertex that corresponds to the first (resp. last) occurrence of $\neg x_i$.

(Here “first” and “last” mean the same thing as in the construction of \mathcal{G}_{var} (Lemma 4.21): they are defined with respect to an arbitrary order on the set of clauses. To build \mathcal{G}_{var} , each variable must occur at least once positively and once negatively; this can be assumed without loss of generality for instances of $\forall\exists$ -CNF-SAT. It is possible that $u_i = v_i$, but for $i \neq j$, we have $u_i \neq v_j$.) We take

$$\mathfrak{P} = \left\{ \{(w, u_i), (w, u_i^-)\} \mid i \in \{1, \dots, m\}, w \in \begin{cases} \{s\} & \text{when } i = 1 \\ \{v_{i-1}, v_{i-1}^-\} & \text{otherwise} \end{cases} \right\}$$

The reader may check that all edges that appear in \mathfrak{P} are indeed edges in \mathcal{G}_{var} . Furthermore, it follows from the definition that the pairs are disjoint and any two paired edges have the same source. Therefore, $(\mathcal{G}_{\text{var}}, \mathfrak{P})$ is a paired digraph in the sense of Definition 4.25.

The point is that, as illustrated in Figure 14, for any switching \mathcal{S} of $(\mathcal{G}_{\text{var}}, \mathfrak{P})$, exactly one of v_m and v_m^- is reachable from s , and the path is unique – let us call it $\pi_{\mathcal{S}}$. Each switching thus induces an assignment $\{x_1, \dots, x_m\} \rightarrow \{\text{true}, \text{false}\}$, which assigns x_i to **false** when $\pi_{\mathcal{S}}$ visits the vertices associated to x_i (among which are u_i and v_i), and to **true** otherwise (in which case the path necessarily visits u_i^- and v_i^-); and the paths from t to s in the switching correspond to extending this assignment with values for the other (existential) variables. Moreover, this map from switchings to assignments of the universal variables is surjective (more precisely, each of the 2^m assignments is induced by exactly 2^{m-1} switchings among the total of 2^{2m-1} possible switchings).

Let us consider next the correspondence given by Lemma 4.21 between the paths from t to s and the assignments of all variables (both universal and existential). One can see that given such a path ρ and a switching \mathcal{S} , the following are equivalent:

- the edges of ρ exist in \mathcal{S} ;
- $\pi_{\mathcal{S}}$ is a prefix of ρ ;
- the assignment that corresponds to \mathcal{S} (by the above discussion) is the restriction to the universal variables of the one that corresponds to ρ (by Lemma 4.21).

From these observations, one can deduce, by a similar reasoning to the proof of Theorem 4.19, that the given $\forall\exists$ -CNF-SAT instance is positive if and only if, for every switching \mathcal{S} of $(\mathcal{G}_{\text{var}}, \mathfrak{P})$, there exists a path from s to t in \mathcal{G}_{cl} and a path from t to s in \mathcal{S} that do not share any intermediate vertex. To conclude, it suffices to reduce this to the desired problem on paired RB-digraphs by a suitable adaptation of Lemma 4.22, whose precise formulation is left to the reader. \square

Theorem 4.27. *Unprovability in pomset logic is Π_2^{P} -hard.*

Proof. We reduce the problem shown to be Π_2^{P} -hard by Lemma 4.26 to pomset logic unprovability by extending proofification to handle edge pairs. An example is given in Figure 15.

Let $(V_{\mathcal{G}}, R_{\mathcal{G}}, B_{\mathcal{G}}, \mathfrak{P})$ be a paired RB-digraph (we write $\mathcal{G} = (V_{\mathcal{G}}, R_{\mathcal{G}}, B_{\mathcal{G}})$). We assume without loss of generality that for every edge pair $\{(u, v), (u, w)\} \in \mathfrak{P}$, the opposite edges (v, u) and (w, u) are not present in the graph (otherwise, break (v, u) up into a path of length three involving two new vertices, add the middle edge of this path to the matching, and similarly for (w, u)). Let us define the following flat sequent, which is *not* balanced in general (the description deliberately mirrors that of $\Pi(\mathcal{G})$ in Definition 4.15):

$$\Gamma = [C_{u_1} \otimes C_{v_1}, \dots, C_{u_m} \otimes C_{v_m}, D_{e_1}, \dots, D_{e_k}, D'_{p_1}, \dots, D'_{p_l}]$$

where (using again the notation of Definition 2.12):

- $\{e_1, \dots, e_k\}$ is the set of *unpaired* edges, $\{(u_1, v_1), (v_1, u_1), \dots, (u_m, v_m), (v_m, u_m)\} = B_{\mathcal{G}}$, and $\{p_1, \dots, p_l\} = \mathfrak{P}$, all these enumerations being non-repeating;
- the atoms involved in Γ are created as follows:
 - for each $\{u, v\} \subseteq V_{\mathcal{G}}$ such that $(u, v) \in R_{\mathcal{G}}^{\otimes}$ – according to the assumption w.l.o.g. we made above, both (u, v) and (v, u) are then unpaired – we give the names $a_{u,v}$ and $a_{v,u}$ to a fresh pair of *dual* atoms;
 - for each edge $(u, v) \in R_{\mathcal{G}}^{\text{a}}$ which is unpaired, we generate two fresh atoms $a_{u,v}$ and $a_{v,u}$ that are *not* dual;

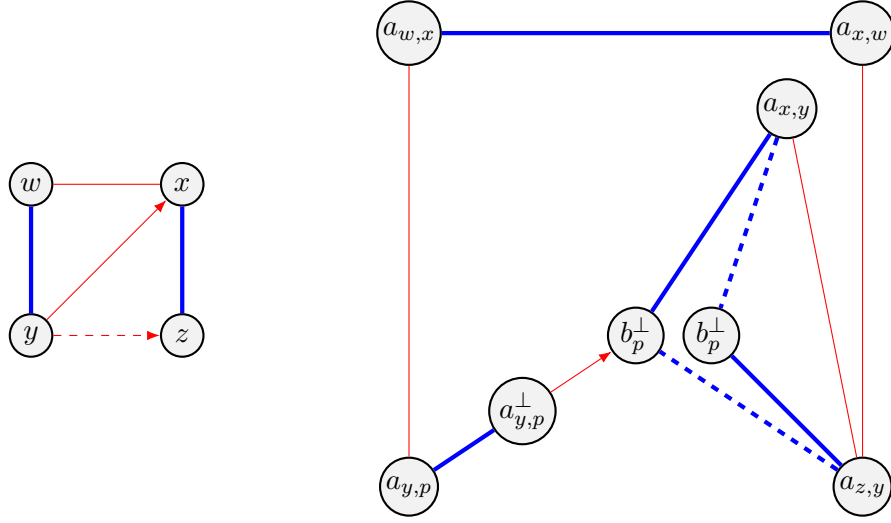


FIGURE 15. Example for the proof of Theorem 4.27.

Left: a paired RB-digraph (compare Figure 11); the two non-matching edges coming out of y are paired, and the dashed edge represents a choice of switching in which it is deleted.

Right: the cographic RB-prenet corresponding to this choice, with $p = \{(y, x), (y, z)\}$ (the dashed blue/bold edges display the matching that would correspond to the other possible choice). Note that we have $a_{x,y} = a_{z,y} = b_p$.

- for each pair $p = \{(u, v), (u, w)\} \in \mathfrak{P}$, we create two fresh atoms $a_{u,p}, b_p$ and we define $a_{v,u} = a_{w,u} = b_p$ (observe however that we do *not* define $a_{u,v}$ nor $a_{u,w}$);
- for $u \in V_G$, $C_u = \bigotimes_v a_{u,v} \wp \bigotimes_p a_{u,p}$ where v (resp. p) ranges over the vertices (resp. pairs) such that $a_{u,v}$ (resp. $a_{u,p}$) is defined according to the above;
- for $(u, v) \in R_G^\triangleleft$, if (u, v) is unpaired, then $D_{(u,v)} = a_{u,v}^\perp \triangleleft a_{v,u}^\perp$;
- finally, for $p = \{(u, v), (u, w)\} \in \mathfrak{P}$, we set $D'_p = \langle a_{u,p}^\perp \triangleleft b_p^\perp \rangle \wp b_p^\perp$.

This sequent Γ that we just defined contains, for each pair $p = \{(u, v), (u, w)\} \in \mathfrak{P}$, exactly two occurrences of b_p , designated as $a_{v,u}$ and $a_{w,u}$, and exactly two occurrences of b_p^\perp in D'_p , that we will also name: $D'_p = \langle a_{u,p}^\perp \triangleleft c_{p^\triangleleft} \rangle \wp c_{p^\wp}$. So each pre-proof of Γ induces a bijection between $\{a_{v,u}, a_{w,u}\}$ and $\{c_{p^\triangleleft}, c_{p^\wp}\}$. Conversely, such a choice of bijection for each edge pair uniquely determines a corresponding axiom linking.

Let ℓ be a pre-proof of Γ . Let $s, t : \mathfrak{P} \rightarrow V_G$ be defined by $\ell(c_{p^\wp}) = a_{t(p), s(p)}$ for all $p \in \mathfrak{P}$. We define the RB-digraph \mathcal{S}_ℓ to be the switching where the edge $(s(p), t(p))$ has been deleted in each pair p . We can reformulate what we observed in the previous paragraph as the fact that $\ell \mapsto \mathcal{S}_\ell$ is a bijection from axiom linkings for Γ to switchings of our paired RB-digraph.

In the cographic RB-prenet $\rho(\Gamma, \ell)$, the vertex for c_{p^\wp} has no incident non-matching edge, so it cannot be involved in any \wp -cycle. Since it is matched with $a_{t(p), s(p)}$, the latter also cannot occur in any \wp -cycle. So the witnesses of incorrectness in $\rho(\Gamma, \ell)$, i.e. its chordless \wp -cycles, must be entirely contained in the induced sub-RB-digraph that excludes c_{p^\wp} and $a_{t(p), s(p)}$ for all $p \in \mathfrak{P}$. Thanks to the correspondence between pseudo-subformulas and

induced subgraphs (Proposition 3.5), one can write (an isomorphic copy of) this sub-RB-digraph as $\rho(\Gamma', \ell')$ where Γ' is obtained from Γ by substituting the aforementioned atom occurrences by \mathbb{I} , and ℓ' is a restriction of ℓ ; so (Γ, ℓ) is incorrect iff (Γ', ℓ') is.

The key property, that one can check from the definitions, is that Γ' is balanced and equal up to atom renaming to the proofification $\Pi(\mathcal{S}_\ell)$ (note for instance that if $(u, v) \in p \in \mathfrak{P}$ and $(u, v) \in R_{\mathcal{S}_\ell}$ then the formula D'_p in Γ becomes $D_{(u,v)}$ in $\Pi(\mathcal{S}_\ell)$ after substituting $c_{p\mathfrak{P}}$ by \mathbb{I}). Thus, by Theorem 4.17, (Γ, ℓ) is incorrect if and only if \mathcal{S}_ℓ contains an \mathfrak{a} -cycle. By definition of provability, and using the surjectivity of $\ell \mapsto \mathcal{S}_\ell$, it follows that Γ is unprovable if and only if every switching of the initial RB-graph admits an \mathfrak{a} -cycle. According to Lemma 4.26, it is **NP**-hard to test the latter condition given the input Γ . Since Γ can be computed from our original paired RB-digraph in polynomial time, this concludes the proof. \square

Proposition 4.24 and Theorem 4.27 together are what it means, by definition, for provability in pomset logic to be Σ_2^P -complete. Thus, we have established Theorem 4.23.

5. BACK TO THE SEQUENT CALCULUS

In this section we come back to the problem of finding a sequent calculus for **BV** and pomset logic, as this problem was the starting point for much of the research done on these two logics. The current state of the art on this topic is:

- (1) Retoré [Ret93] presented a sequent calculus for pomset logic, for which he could show soundness, but not completeness and not cut elimination.¹⁹
- (2) This difficulty motivated Gugliemi [Gug07] to develop system **BV**, which has a cut elimination proof in deep inference, but not in the sequent calculus. Then Tiu [Tiu06] has shown that “deepness” is necessary for **BV** and therefore there cannot be a sequent calculus for **BV** (see also Remark 4.12).
- (3) However, the formulas used by Tiu [Tiu06] to defy the sequent calculus can be proved in the cut-free version of Retoré’s [Ret93] sequent calculus, using the *entropy* rule.
- (4) In Remark 4.12 we also observed that our complexity results pose a much harder obstacle to a sequent calculus for pomset logic than the mere need of “deepness”.
- (5) But recently, Slavnov [Sla19] presented a cut-free sequent calculus that is sound and complete for pomset logic.

This is, of course, confusing, as it seems contradictory. Does (3) mean that Tiu’s [Tiu06] result is wrong? Does (5) mean that Retoré [Ret93] just did not try hard enough and the problem is solved? Does it mean that our complexity result about the Σ_2^P -completeness of pomset logic is wrong or that **NP** = **coNP**?

The answer to all these questions is, of course, “No”. And the purpose of this section is to clarify the confusion and work out the subtleties of possible sequent calculi for pomset logic and **BV**. More precisely we present the following results:

- We show that Retoré’s sequent calculus with cut (the variant presented in [Ret21]) is equivalent to **SBV**. It is therefore a sequent calculus for **BV** and not for pomset logic.

¹⁹In fact, in [Ret97a] and [Ret21] two variations of that sequent calculus are proposed, which both have the same problem.

$$\begin{array}{c}
\text{axiom} \frac{}{[a, a^\perp]} \quad \text{dimix} \frac{\Gamma \quad \Delta}{\langle \Gamma; \Delta \rangle} \quad \text{entropy} \frac{\Gamma}{\Gamma'} \text{ (for } \Gamma' \preceq \Gamma) \quad \text{cut} \frac{[\Gamma, A] \quad [A^\perp, \Delta]}{[\Gamma, \Delta]} \\
\text{\textcircled{R}}\text{-intro} \frac{\Gamma\{[A, B]\}}{\Gamma\{A \textcircled{R} B\}} \quad \text{\textless}\text{-intro} \frac{\Gamma\{\langle A; B \rangle\}}{\Gamma\{A \textless B\}} \quad \text{\textcircled{\otimes}}\text{-intro} \frac{[\Gamma, A] \quad [\Delta, B]}{[\Gamma, \Delta, A \textcircled{\otimes} B]}
\end{array}$$

FIGURE 16. Retoré’s sequent calculus, as presented in [Ret21, Figure 1]

- We present a formula²⁰ that is provable in BV but not in the cut-free version of Retoré’s [Ret93] sequent calculus. Therefore, that sequent calculus does not admit cut elimination.
- We refine Tiu’s [Tiu06] argument about the need for “deepness” in BV such that Retoré’s *entropy* rule is no longer enough to prove the formulas used in the argument.
- We use our results from the previous section to make a complexity-theoretic argument to show that a “standard” sequent calculus for pomset logic is impossible.
- Finally, we discuss Slavnov’s [Sla19] sequent calculus and exhibit how he circumvents this complexity-theoretic obstacle.

5.1. Retoré’s Sequent Calculus with Cuts is Equivalent to BV. Recall that a pomset logic sequent can be seen as a \otimes -free generalized formula over a set of formula occurrences (see Proposition 2.8). Hence, we can define the graph of a sequent Γ as in Definition 2.17. Let $\mathcal{G}_\Gamma = (V_\Gamma, R_\Gamma)$ and $\mathcal{G}_{\Gamma'} = (V_{\Gamma'}, R_{\Gamma'})$ be the graphs of sequents Γ and Γ' , respectively. Then both are series-parallel orders (see Proposition 2.31), and we define

$$\Gamma' \preceq \Gamma \quad \text{iff} \quad V_{\Gamma'} = V_\Gamma \quad \text{and} \quad R_{\Gamma'} \subseteq R_\Gamma \quad (5.1)$$

i.e., both sequents contain the same formula occurrences, and the order induced by Γ' is contained in the order induced by Γ . We are now ready to discuss Retoré’s sequent calculus, shown in Figure 16.²¹

The $\textcircled{\otimes}$ -introduction rule and the \textless -introduction rule simply express the correspondence between the structural/sequent level connectives and the logical/formula level connectives. Note that the $\textcircled{\otimes}$ -introduction rule and the cut rule can only be applied in flat contexts (but Γ and Δ can be non-flat sequents). The standard mix-rule is derivable using the dimix- and entropy-rules.

Before embarking on the equivalence of this sequent calculus with BV, let us illustrate the added power of cuts. For this purpose, we recall an example sequent considered by Retoré and the second author:

$$[a \otimes (c \textless b^\perp), a^\perp \textless f, c^\perp \textless d^\perp, d \otimes (e^\perp \textless f^\perp), e \textless b] \quad (5.2)$$

The corresponding formula of the sequent above has been shown to be provable in BV in Figure 5, and its cographic RB-prenet may be found in [Ret21, Figure 10].

Proposition 5.1 (Retoré & Straßburger [Ret21, Proposition 5]). *The sequent in (5.2) is not provable in Retoré’s sequent calculus without cuts.*

²⁰This formula has already been presented in [Ret21].

²¹In this calculus, which is taken from [Ret21] sequents define SP-orders. In [Ret93], sequent can be arbitrary orders, and in [Ret97a], the entropy rule is missing.

$$\begin{array}{c}
\text{cut} \frac{\frac{\pi_1 \text{ (below)}}{[a \otimes (c \triangleleft b^\perp), a^\perp \triangleleft f, c^\perp \triangleleft (e^\perp \triangleleft f^\perp), e \triangleleft b]}{\quad} \quad \frac{\pi_2 \text{ (below)}}{[c \triangleleft (e \triangleleft f), c^\perp \triangleleft d^\perp, d \otimes (e^\perp \triangleleft f^\perp)]}}{[a \otimes (c \triangleleft b^\perp), a^\perp \triangleleft f, c^\perp \triangleleft d^\perp, d \otimes (e^\perp \triangleleft f^\perp), e \triangleleft b]} \\
\\
\text{where } \pi_1 = \frac{\text{axiom} \frac{\text{axiom} \frac{\text{axioms + dimix}}{\langle [c, c^\perp]; [e, e^\perp]; [b, b^\perp] \rangle} \text{entropy} \frac{\langle [c, b^\perp], \langle c^\perp; e^\perp \rangle; \langle e; b \rangle \rangle}{[c \triangleleft b^\perp, \langle c^\perp; e^\perp \rangle; \langle e; b \rangle]} \triangleleft\text{-intro} \frac{[a, a^\perp]}{[a \otimes (c \triangleleft b^\perp), a^\perp, \langle c^\perp; e^\perp \rangle, \langle e; b \rangle]} \otimes\text{-intro} \frac{[a \otimes (c \triangleleft b^\perp), a^\perp, \langle c^\perp; e^\perp \rangle, \langle e; b \rangle]}{[a \otimes (c \triangleleft b^\perp), a^\perp, \langle c^\perp; e^\perp \rangle, \langle e; b \rangle]} \text{axiom} \frac{[f, f^\perp]}{[f, f^\perp]} \text{dimix} \frac{\langle [a \otimes (c \triangleleft b^\perp), a^\perp, \langle c^\perp; e^\perp \rangle, \langle e; b \rangle]; [f, f^\perp] \rangle}{[a \otimes (c \triangleleft b^\perp), \langle a^\perp; f \rangle, \langle c^\perp; e^\perp \rangle, \langle e; b \rangle]} \text{entropy} \frac{[a \otimes (c \triangleleft b^\perp), a^\perp \triangleleft f, c^\perp \triangleleft (e^\perp \triangleleft f^\perp), e \triangleleft b]}{[a \otimes (c \triangleleft b^\perp), a^\perp \triangleleft f, c^\perp \triangleleft (e^\perp \triangleleft f^\perp), e \triangleleft b]} \triangleleft\text{-intro} \times 4 \\
\\
\text{and } \pi_2 = \frac{\text{axiom} \frac{\text{axiom} \frac{[e \triangleleft f, e^\perp \triangleleft f^\perp]}{[e \triangleleft f, d^\perp, d \otimes (e^\perp \triangleleft f^\perp)]} \text{axiom} \frac{[d, d^\perp]}{[d, d^\perp]} \otimes\text{-intro} \frac{[c, c^\perp]}{[c, c^\perp]} \text{dimix} \frac{\langle [c, c^\perp]; [e \triangleleft f, d^\perp, d \otimes (e^\perp \triangleleft f^\perp)] \rangle}{\langle [e; e \triangleleft f], \langle c^\perp; d^\perp \rangle, d \otimes (e^\perp \triangleleft f^\perp) \rangle} \text{entropy} \frac{[c \triangleleft (e \triangleleft f), c^\perp \triangleleft d^\perp, d \otimes (e^\perp \triangleleft f^\perp)]}{[c \triangleleft (e \triangleleft f), c^\perp \triangleleft d^\perp, d \otimes (e^\perp \triangleleft f^\perp)]} \triangleleft\text{-intro} \times 2}{}
\end{array}$$

FIGURE 17. Proof of the formula from Figure 5 in Retoré's sequent calculus with cuts.

The reason for this lies in the fact that in order to have a cut-free sequent proof, the \otimes -rule has to be applied to one of the two \otimes -formulas eventually. But the \otimes -rule is too weak to split the context correctly.

However, with cut, the sequent is provable, as shown in Figure 17. We have therefore shown the following.

Corollary 5.2. *Cut-elimination does not hold for Retoré's sequent calculus.*

Let us now establish the equivalence between SBV and the sequent calculus in Figure 16. For this, we resort again to SBVu (see Proposition 2.68). We begin by showing that every formula that is provable in SBVu is also provable in Retoré's sequent calculus with cut.

Lemma 5.3. *Let $[A, B]$ and $[C, D]$ be provable sequents. Then $[A \wp C, B \otimes D]$ and $[A \triangleleft C, B \triangleleft D]$ are also provable.*

Proof. The following derivations work:

$$\begin{array}{c}
\otimes\text{-intro} \frac{[A, B] \quad [C, D]}{[A, C, B \otimes D]} \\
\wp\text{-intro} \frac{[A, C, B \otimes D]}{[A \wp C, B \otimes D]} \\
\\
\text{dimix} \frac{[A, B] \quad [C, D]}{\langle [A, B]; [C, D] \rangle} \\
\text{entropy} \frac{\langle [A, C], \langle B; D \rangle \rangle}{\langle [A, C], \langle B; D \rangle \rangle} \\
\triangleleft\text{-intro} \frac{[A \triangleleft C, \langle B; D \rangle]}{[A \triangleleft C, \langle B; D \rangle]} \\
\triangleleft\text{-intro} \frac{[A \triangleleft C, \langle B; D \rangle]}{[A \triangleleft C, B \triangleleft D]}
\end{array}$$

For the derivation on the right, the validity of the entropy rule depends on an inclusion between series-parallel orders which can be mechanically verified. \square

Lemma 5.4 (Non-atomic identity). *For any formula A , the sequent $[A^\perp, A]$ is provable.*

Lemma 5.5. *Let A and B be formulas such that $[A^\perp, B]$ is provable, and $S\{\cdot\}$ be a context. Then $[S\{A\}^\perp, S\{B\}]$ is provable.*

Proof of Lemmas 5.4 and 5.5. We prove that the following holds for any formula A :

- (i) $[A^\perp, A]$ is provable;
- (ii) for any formulas B and C such that $[B^\perp, C]$ is provable and any $\odot \in \{\otimes, \wp, \triangleleft\}$, the sequents $[(A \odot B)^\perp, A \odot C]$ and $[(B \odot A)^\perp, C \odot A]$ are provable.

Lemma 5.4 corresponds to item (i), while item (ii) entails Lemma 5.5 by a routine induction on the context $S\{\cdot\}$. The following observations suffice to prove (i) and (ii) simultaneously by induction on A (invoking the axiom rule for the base case):

- for any given A , (i) implies (ii) as a direct consequence of Lemma 5.3;
- if A satisfies (i) and A' satisfies (ii), then by applying (ii) to A' with $B = C = A$, we see that for any $\odot \in \{\otimes, \wp, \triangleleft\}$, the formula $A \odot A'$ satisfies (i). \square

Lemma 5.6. *Let $r \frac{A}{B}$ be an instance of an inference rule in SBVu. with Then $[A^\perp, B]$ is provable in Retoré's sequent calculus without cuts.*

Proof. Let us consider for instance the switch rule. We start by considering the case with atomic formulas a, b, c and a trivial context ($S\{\cdot\} = \{\cdot\}$):

$$s \frac{[a \wp c] \otimes b}{(a \otimes b) \wp c}$$

We prove the corresponding sequent (for the switch rule, the derivation involves only flat sequents and can therefore be carried out in MLL+mix):

$$\frac{\frac{\frac{[a^\perp, a] \quad [b^\perp, b]}{[a^\perp, b^\perp, a \otimes b]} \quad [c^\perp, c]}{[a^\perp \otimes c^\perp, b^\perp, a \otimes b, c]}}{[a^\perp \otimes c^\perp, b^\perp, (a \otimes b) \wp c]}}{[(a^\perp \otimes c^\perp) \wp b^\perp, (a \otimes b) \wp c]}$$

Thanks to Lemma 5.4, we can substitute formulas for atoms: for any A, B and C , the sequent $[(A^\perp \otimes C^\perp) \wp B^\perp, (A \otimes B) \wp C]$ can be proved using the above derivation where axioms are replaced by appropriate subproofs of $[A^\perp, A]$, $[B^\perp, B]$ or $[C^\perp, C]$. Then, using Lemma 5.5, we may conclude that $[S\{[A \wp C] \otimes B\}^\perp, S\{(A \otimes B) \wp C\}]$ is provable for any context $S\{\cdot\}$.

This argument applies to all inference rules (except $\text{ai}^\circ\downarrow$, which has no premise, and therefore does not fit the pattern in the lemma statement). For each of these rules, it therefore suffices to treat the case with atomic formulas and trivial context.

For the \equiv' -rule, we can assume without loss of generality that each instance is an application of exactly one of the equalities in (2.5) (as each general instance of the \equiv' -rule can be replaced by a finite sequence of these special instances).

We have thus reduced the desired conclusion to a bounded search for cut-free proofs that we leave to the reader. \square

Theorem 5.7. *If a formula is provable in SBVu then it is also provable in Retoré’s sequent calculus with cuts.*

Proof. A proof of a formula A in SBVu must have the form

$$\begin{array}{c} \text{ai}^\circ\downarrow \frac{}{a \wp a^\perp} \\ r_1 \frac{}{B_1} \\ r_2 \frac{}{\vdots} \\ r_n \frac{}{B_n} \end{array}$$

where $B_n = A$; we also define $B_0 = a \wp a^\perp$. For $i \in \{1, \dots, n\}$, the sequent $[B_{i-1}^\perp, B_i]$ is provable in Retoré’s sequent calculus by Lemma 5.6. Furthermore, $a \wp a^\perp$ also has a sequent calculus proof (an axiom rule followed by a \wp -intro). By composing all these sequent proofs with the cut rule, one gets a proof of A . \square

For the converse, observe that the axiom, dimix and \wp/\triangleleft -intro rules are easy to simulate in BVu. The treatment of \otimes -intro is as usual (see, e.g., [Str03b, Section 3.3.] or [Gug07, §5]), and cut is a \otimes -intro followed by i^\uparrow (which is admissible in SBVu). Furthermore, it is an immediate consequence of Theorem 2.76 that SBVu can also simulate the entropy rule. This is enough to prove the following theorem:

Theorem 5.8. *If a sequent Γ is provable in Retoré’s sequent calculus with cuts, then a formula A that corresponds (see Definition 2.9) to Γ is provable in SBVu.*

Corollary 5.9. *Retoré’s sequent calculus with cuts is equivalent to BV.*

5.2. A Refinement of Tiu’s Argument. In [Tiu06], Tiu presents a sequence of formulas S_0, S_1, S_2, \dots with the following properties:

- (1) For each n , the formula S_n is provable in BV.
- (2) In order to prove S_n , a subformula at depth $2n$ has to be accessed first.

From this it follows, that there can be no *shallow* (i.e., all inference rules have a fixed maximum depth) cut-free proof system equivalent to BV. As most standard sequent calculi are shallow in that sense, the argument can be used to claim that there cannot be a cut-free sequent calculus for BV. However, Tiu’s formulas also have the property

- (3) For each n , the formula S_n is \otimes -free.

Since every S_n contains only the \wp and \triangleleft connectives, it can be proved in Retoré’s calculus by first applying the \wp and \triangleleft rules to transform the formula into a sequent with only atomic formulas. This sequent can then be derived from the axioms by only dimix instances and a single instance of the entropy rule.

This is not a contradiction, as the entropy rule is a *deep* rule (i.e., not shallow) in the sense above. However, it raises the question whether we can have a refinement of Retoré’s sequent calculus that is complete for BV and obeys cut elimination, as this is no longer ruled out by Tiu’s argument.

What we will show next is a sequence of formulas R_0, R_1, R_2, \dots , following the spirit of Tiu's construction, having properties (1) and (2) above, but not (3). Consequently, for proving them, a \otimes -subformula has to be accessed at arbitrary depth, without the possibility of globally splitting the context. This entails that a proper deep inference system is indeed needed, and a proof system in the sequent calculus layout is insufficient.

We start with the index set $I = \{0, 1\}^*$, i.e., the set of all finite words with the symbols 0 and 1. Then, our formulas are built from the propositional variables $\{a, b, c, y, z\} \times I$, written as a_i with $i \in I$.

For an index $i \in I$ and two formulas A and B , we define now inductively the formula $\xi_n(i, A, B)$ for each $n \in \mathbb{N}$:

$$\begin{aligned} \xi_0(i, A, B) &= ([a_i \wp b_i \wp A] \otimes y_i) \wp \langle y_i^\perp \triangleleft c_i \rangle \wp \langle a_i^\perp \triangleleft z_i \rangle \wp (z_i^\perp \otimes [b_i^\perp \wp c_i^\perp \wp B]) \\ \xi_{n+1}(i, A, B) &= (\xi_n(i0, a_i, b_i \wp A) \otimes y_i) \wp \langle y_i^\perp \triangleleft c_i \rangle \wp \langle a_i^\perp \triangleleft z_i \rangle \wp (z_i^\perp \otimes \xi_n(i1, b_i^\perp, c_i^\perp \wp B)) \end{aligned}$$

We now define $R_n = \xi_n(0, \mathbb{I}, \mathbb{I})$.²² These formulas have the following properties:

Claim 5.10. For each n , the formula R_n is provable in BV.

Proof. For every $n \in \mathbb{N}$ and $i \in I$, we have a derivation
$$\frac{A \wp B}{\xi_n(i, A, B)}$$
. This is constructed in

the same way as in the proof of Lemma 7.4 in [Tiu06], using a derivation that is similar to the one in Figure 5. \square

Claim 5.11. The formulas R_n are not provable in Retoré's sequent calculus without cuts.

Proof. The formula R_0 is a variation of the corresponding formula of the sequent (5.2), and the same argument applies. \square

Finally, we have:

Claim 5.12. A deep proof system is needed to prove the formulas R_n .

Proof. This is proved by the same argument as in [Tiu06]. In order to prove R_n , a \otimes at depth $2n$ has to be accessed first, in order to remove the variable y_i (or z_i) with an atomic interaction. \square

This strengthens Tui's argument by using formulas involving also a \otimes , showing that rules like entropy are not enough to obtain a cut-free sequent style calculus for BV.

5.3. A Complexity-Theoretic Obstacle to Sequent Calculi for Pomset Logic. Let us now turn to pomset logic. We propose here a similar (and somewhat informal) complexity-theoretic explanation for why a cut-free sequent calculus for pomset logic is difficult to find, similar to Remark 4.12.

As described in [Ret21, Section 5], the origins of pomset logic in the coherence space semantics of linear logic suggest that \triangleleft is meant to be a *multiplicative* connective, just like the multiplicative conjunction \otimes and the multiplicative disjunction \wp (which give their names to the fragments MLL and MLL+mix of linear logic). Therefore, we would like to leverage some proof-theoretic property related to multiplicative connectives. Recall that

²²Note that the only difference between our ξ_n and Tiu's α_n is the addition of the variables y_i and z_i via the \otimes .

the standard sequent calculus rules for \otimes and for its *additive* counterpart $\&$ correspond to two possible introduction rules for the conjunction \wedge in classical logic, respectively:

$$\frac{\vdash \Gamma, A \quad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \wedge B} \quad \text{and} \quad \frac{\vdash \Gamma, A \quad \vdash \Gamma, B}{\vdash \Gamma, A \wedge B}$$

The difference is that the multiplicative rule *splits the context of the conclusion into disjoint parts among the premises* while the additive rule copies the same context in both premises. The same kind of management of the context occurs in the rules for the so-called “generalized multiplicative connectives” in the French-Italian linear logic tradition [DR89, AM20]. This leads us to the following definition.

Definition 5.13. A *multiplicative introduction rule* for an n -ary connective Φ is a rule whose instances obey the following property: there exist formulas A_1, \dots, A_n and a multiset of formulas M such that

- the multiset of formulas occurring in the premises — i.e. the sum of the multisets of formulas in each premise — is equal to²³ $\{A_1, \dots, A_n\} + M$;
- the multiset of formulas occurring in the conclusion equals $\Phi(A_1, \dots, A_n) + M$.

A *multiplicative structural rule* is one in which the premises (taken together) and the conclusion have the same formulas, taking multiplicity into account; this is an equality of multisets similar to the above.

A sequent calculus is *multiplicative* when all its rules are either an axiom rule, a multiplicative introduction rule or a multiplicative structural rule.

For this to make sense, there must be a way to associate to a sequent its multiset of formulas. This works both for flat sequents — which are already multisets — and for ordered sequents, and other kinds of generalized sequents are conceivable. The standard MLL+mix sequent calculus, as well as Retoré’s calculus in the previous section and Slavnov’s calculus in the next one, are all multiplicative.

The relevance of this notion to us is a consequence in the spirit of proof complexity:

Claim 5.14. In a proof in a multiplicative sequent calculus, the total number of introduction rules is at most the total size of the formulas in the sequent being proved.

In many cases, there will also be a bound on the structural rules in proofs. For instance, the number of uses of multiplicative structural rules that require two or more premises, such as the mix rule, is also linearly bounded by the size of the conclusion formulas. Thus the following informal principle: a multiplicative sequent calculus with “reasonable” structural rules admits proofs with a polynomial number of inference rules.

We also expect the correctness of these proofs to hinge only on the local validity of their inferences (unlike proof nets, whose global correctness criterion makes them hard to check). According to a similar reasoning as Remark 4.12, then, if such a multiplicative calculus were to capture pomset logic, it would be impossible to verify its inference rules in time polynomial in the size of the formulas, unless $\mathbf{NP} = \mathbf{coNP}$. This is arguably a strong restriction on the design of sequent calculi for pomset logic, such as the calculus in the next section.

²³We abusively use the notation $\{\dots\}$ for a multiset rather than a set here.

5.4. A Reconstruction of Slavnov’s Calculus. Let us now revisit the sound and complete proof system of [Sla19] for pomset logic in light of our above remarks. It uses *decorated sequents*, which are flat sequents (multisets of formulas) endowed with additional “decorations” (analogously to how Retoré’s sequents can be seen as flat sequents decorated with series-parallel orders). The point is that in Slavnov’s sequents, these decorations (defined in Definition 5.15 below) take up most of the space; their size may be *exponentially bigger* than the number of formulas in the sequent. This provides a good reason for inference rule checking to be superpolynomial in the total size of formulas — this is indeed necessary since this sequent calculus is, as we shall see, multiplicative and “reasonable” in the above sense.

For the sake of completeness, we describe briefly here Slavnov’s system, specialized to pomset logic. The paper [Sla19] also introduces and deals with a related but different extension of MLL called “semicommutative MLL”, and it derives its sequent calculus for pomset logic from the one for semicommutative MLL. Our exposition avoids this detour. We also aim at giving a high-level idea of what makes the system work. We decompose our presentation of the system into three “levels” (inspired by the two-level treatment of [Sla19]).

First level: preproofs on flat sequents. First, consider the “old-fashioned” calculus on flat sequents whose rules are as follows:

$$\text{axiom } \frac{}{[a, a^\perp]} \quad \text{mix } \frac{\Gamma \quad \Delta}{[\Gamma, \Delta]} \quad \otimes\text{-intro } \frac{[\Gamma, A] \quad [\Delta, B]}{[\Gamma, \Delta, A \otimes B]} \quad \odot\text{-intro } \frac{[\Gamma, A, B]}{[\Gamma, A \odot B]} \text{ for } \odot \in \{\wp, \triangleleft\}$$

In other words, this system extends the usual cut-free sequent calculus for MLL+mix with an introduction rule for \triangleleft which is exactly the same as for \wp . Let us call *Slavnov pre-proofs* the derivation trees using these rules. Obviously, some formulas that are not valid in pomset logic, such as $(a \otimes b) \wp (a^\perp \triangleleft b^\perp)$, may admit Slavnov pre-proofs.

An important observation is that *there is a canonical map from Slavnov pre-proofs to tree-like RB-prenets* that preserves the conclusion sequent. It can be defined inductively by interpreting each of the inference rules as an operation on proof nets in the obvious way; for instance the \otimes -intro rule corresponds to taking the union of two RB-prenets and adding a gadget for the newly added \otimes connective (cf. Figure 2).

The goal of the “second level” of the proof system will then be to filter out Slavnov pre-proofs that translate into correct tree-like RB-nets.

(For MLL (resp. MLL+mix) sequent calculus proofs, i.e. Slavnov pre-proofs without the \triangleleft -intro and mix rules (resp. the \triangleleft -intro rule), it is well-known that the result of the translation is a correct MLL (resp. MLL+mix) proof net: in those cases, it corresponds to a “desquentialization” operation whose idea goes back to Girard’s original paper [Gir87].)

Second level: decorations with “multi-reachability” information. At this stage, it is natural to add data that keeps track of paths in RB-prenets, in order to reflect the correctness criterion.

Definition 5.15. A *decoration* on a flat sequent Γ is a set \mathfrak{d} of sets of ordered pairs of formula occurrences. The pair (Γ, \mathfrak{d}) is then called a *decorated sequent*.

To each Slavnov pre-proof π of a flat sequent Γ , we associate a decoration \mathfrak{d}_π — and thus the decorated sequent $\mathfrak{S}_\pi = (\Gamma, \mathfrak{d}_\pi)$ — as follows: first translate it into a tree-like RB-prenet \mathcal{G} , and then let the set $\{(A_1, B_1), \dots, (A_n, B_n)\} \in \mathfrak{d}_\pi$ whenever there exists a family of *pairwise disjoint* \wp -paths $(P_i)_{i \in \{1, \dots, n\}}$ such that for each i the path P_i goes from the conclusion vertex of \mathcal{G} corresponding to A_i to the vertex corresponding to B_i .

The key property is now:

Claim 5.16. The decorated sequent corresponding to a Slavnov-pre-proof is entirely determined by the last rule and the decorated sequents corresponding to the sub-pre-proofs of the premises.

For instance, there exists a function \mathfrak{F}_\otimes such that given a proof π equal to

$$\otimes\text{-intro} \frac{\frac{\pi_1}{[\Gamma, A]} \quad \frac{\pi_2}{[\Delta, B]}}{[\Gamma, \Delta, A \otimes B]}$$

we have $\mathfrak{S}_\pi = \mathfrak{F}_\otimes(\mathfrak{S}_{\pi_1}, \mathfrak{S}_{\pi_2}, A, B)$.²⁴ To see why this claim holds, observe that a family of disjoint \mathfrak{a} -paths between conclusions in the tree-like RB-prenet corresponding to π consists of the union of:

- such a family in the prenet for π_1 , not touching A ;
- such a family in the prenet for π_2 , not touching B ;
- either the empty set, or a singleton containing one of the following:
 - an \mathfrak{a} -path between (the vertex for) some formula occurrence in Γ and some other in Δ going through the RB-tree gadget for $A \otimes B$;
 - an \mathfrak{a} -path from some vertex from either Γ or Δ to the conclusion vertex for $A \otimes B$;
 - the reverse of either of the above two possibilities.

An explicit expression for \mathfrak{F}_\otimes can be obtained by reasoning along these lines. Similar (simpler) analyses can be carried out for the other connectives, and for the axiom and mix rules.

This allows us to lift each of the previous inference rules on flat sequents to a rule on decorated sequents, for example the decorated version of \otimes -intro would be

$$\frac{\mathfrak{S} \quad \mathfrak{S}'}{\mathfrak{F}_\otimes(\mathfrak{S}, \mathfrak{S}', A, B)} \quad \text{where } A \text{ (resp. } B) \text{ is a formula occurrence in } \mathfrak{S} \text{ (resp. } \mathfrak{S}')$$

These decorated inference rules can be read as a proof system, and we have:

Claim 5.17. The derivation trees generated by those rules are exactly the ones that can be obtained in the following way: start from a Slavnov pre-proof (with flat sequents) and for each node, replace its value by \mathfrak{S}_π where π is the sub-pre-proof rooted at that node.

Let us call these derivation trees *decorated pre-proofs*.

Third level: side condition using the decorations. At this point, we have obtained a new proof system that, in the end, proves the same flat sequents as the former Slavnov pre-proofs. So it is still unsound with respect to pomset logic. To remedy that, it remains to leverage the additional “multi-reachability” information provided by the decorations (in fact, we use only reachability by a single \mathfrak{a} -path between two formulas).

Claim 5.18. A decorated pre-proof translates to a correct tree-like RB-net if and only if we have $\{(B, A)\} \notin \mathfrak{d}$ for every instance of a \triangleleft -intro rule in it of the form below:

$$\frac{([\Gamma, A, B], \mathfrak{d})}{([\Gamma, A \triangleleft B], \mathfrak{d}')} \quad \text{where } \mathfrak{d} \text{ is a pomset}$$

²⁴Strictly speaking, the two last arguments point to formula occurrences in the conclusions of π_1, π_2 .

Proof. In the inductive translation of Slavnov pre-proofs to prenets, if the last rule is any other than \triangleleft -intro, then the sub-pre-proofs for its premises are all mapped to correct nets if and only if the translation of the whole pre-proof is itself a correct net. In fact, this is precisely why the desequentialization of $\text{MLL} + \text{mix}$ sequent proofs into proof nets is sound, a well-known fact. However, for a \triangleleft -intro rule (using the above notations), the corresponding operation on tree-like RB-prenets may create new \ae -cycles. From the shape of the RB-tree gadget associated to \triangleleft (cf. Figure 2) one can see that such new cycles can only be composed of the directed edge of this gadget from A to B plus an \ae -path from B to A in the prenet for the premise. The existence of the latter is precisely equivalent to $\{(B, A)\} \in \mathfrak{d}$. \square

Note that the assumption in the above claim only consists in purely local “side conditions” on inference rules, hence:

Definition 5.19. A *decorated proof* is a derivation tree in the system whose inference rules are those of decorated pre-proofs except that the \triangleleft -intro rule is subject to a *side condition*: it can only be applied to $([\Gamma, A, B], \mathfrak{d})$ (to introduce $A \triangleleft B$) when $\{(B, A)\} \notin \mathfrak{d}$.

The “if” part of Claim 5.18 can then be rephrased as the *soundness of the decorated proof system* for pomset logic.

Let us also sketch briefly a *completeness* argument. Given a correct pomset logic proof net \mathcal{G} , first consider a copy \mathcal{G}' where every \triangleleft has been turned into \ae . This is still a correct net (replacing \triangleleft by \ae removes an edge, so it can destroy \ae -cycles, not create them), in fact \mathcal{G}' is an $\text{MLL} + \text{mix}$ proof net. There exists an $\text{MLL} + \text{mix}$ sequent proof whose inductive translation (desequentialization) is \mathcal{G}' — this is the *sequentialization theorem* (a result involving non-trivial combinatorics, originating in [Gir87]; see [Ret03, Ngu20] for a discussion of its “equivalence” with earlier results in mainstream graph theory). Next, in this sequentialization, replace the relevant occurrences of \ae by \triangleleft ; we obtain a Slavnov pre-proof (since in this system the rule for \triangleleft is the same as the $\text{MLL} + \text{mix}$ rule for \ae) which lifts uniquely to a decorated pre-proof. Finally, one must check that the latter satisfies the side conditions; this comes from the correctness of the pomset logic proof net \mathcal{G} that we started with, plus the “only if” part of Claim 5.18.

6. CONCLUSION

In the first paper of this series [Gug07], Guglielmi announced the task of the present one in the following way:

It is still open whether the logic in this paper, called BV, is the same as pomset logic. We conjecture that it is actually the same logic, but one crucial step is still missing, at the time of this writing, in the equivalence proof. This paper is the first in a planned series of three papers dedicated to BV. [...] In the third part, some of my colleagues will hopefully show the equivalence of BV and pomset logic, this way explaining why it was impossible to find a sequent system for pomset logic.

Surprisingly, the hoped-for equivalence turned out to be false; in Section 3, we exhibited an explicit formula provable in pomset logic, but not in BV. What first led us to seek such a counter-example was the discovery of the complexity-theoretic hardness results of Section 4, according to which the conjectured equivalence would have implied $\mathbf{NP} = \mathbf{coNP}$. This, plus Slavnov’s recent system [Sla19], put into question the established narrative about the

impossibility of sequent calculi for pomset logic, so we revisited this topic in Section 5 (and showed in passing that an old sequent calculus with cuts was in fact equivalent to BV).

6.1. Related topics. As we hope that this paper may serve as a reference for readers who wish to get acquainted with BV and pomset logic (hence the lengthy Section 2), we will broadly survey here some works that are connected to these two systems, without limiting ourselves to provability or complexity-theoretic aspects.

Applications and semantics of BV and pomset logic. One of the first applications of self-dual non-commutativity was Reddy’s Linear Logic Model of State [Red93]. This work, whose ultimate goal is to study mutable state in programming languages, introduces an extension LLMS of intuitionistic²⁵ linear logic with some connectives, one of which is \triangleleft — LLMS comes with a semantics in coherence spaces where the interpretation of \triangleleft coincides with that for pomset logic. The proof system for LLMS is a sequent calculus similar to Retoré’s one (§5.1). It also admits a semantics in Dialectica categories [dP14, §4].

As for the proof nets of pomset logic, they have no known notion of categorical semantics; in fact, in light of the connections between categorical logic and deep inference [Hug04], it might be argued that any presentation of pomset logic as an “initial something-category” would amount to giving a deductive proof system for it. However the same connections make modeling BV categorically a straightforward matter, as has been done in [BPS12] where a new concrete semantics (probabilistic coherence spaces) is also given as an example of BV-category. There have been recent works relating BV and BV-categories to quantum causality [BGI⁺14, SK22].

Finally, let us mention that Retoré and his collaborators have applied pomset logic to mathematical linguistics (see [Ret21, Section 7]). This provides an alternative to the usual approach in *categorical grammars* [MR12] which relies on the another kind of non-commutative logic that we shall cover next.

Other non-commutative variants of linear logic. Linearity and non-commutativity first appeared in the study of typed λ -calculi in the Lambek calculus [Lam58], whose introduction was motivated by the aforementioned linguistic applications. We might thus consider it retrospectively as the first non-commutative logic, even though the *formulas-as-types correspondence*²⁶ between typed λ -calculi and constructive logics was not known at the time. In the Lambek calculus, the order of arguments of a function matters: thus $A \multimap B \multimap C \not\equiv B \multimap A \multimap C$ where \multimap is the linear implication connective.

In a classical linear logic framework, where $A \multimap B$ may be defined as $A^\perp \wp B$, this translates into $A \wp B \not\equiv B \wp A$ — a non-commutativity in the literal sense. This entails the non-commutativity of its dual connective \otimes and we have $(A \otimes B)^\perp = B^\perp \wp A^\perp$. (In contrast, pomset logic keeps \otimes, \wp commutative while adding the new connective \triangleleft , and the self-duality of \triangleleft does *not* permute its arguments.) The standard system with those properties is *cyclic*²⁷

²⁵In the context of linear logic, “intuitionistic” means that the sequents are two-sided with the right side being limited to a single formula. The sequents in [Red93] are of the form $\Gamma \vdash A$ where Γ is what we call an ordered sequent.

²⁶Also known as “proofs-as-programs” or “Curry–Howard” correspondence.

linear logic; see [Yet90] for its sequent calculus and proof nets, [DG04] for a deep inference system and [AM19] for pointers to more recent work on cyclic MLL.

On λ -terms or proof nets, non-commutativity corresponds to a *planarity* condition; to our knowledge, this was first remarked by Girard in [Gir89, Section II.9] just after his discovery of linear logic [Gir87]. For more recent works pursuing such topological ideas, see e.g. [APR05, Abr07, Mel18]. In particular, renewed interest in the non-commutative linear λ -calculus has come from the discovery of a deep connection with the combinatorics of planar maps [ZG15], including bijective and enumerative aspects.

The aforementioned works consider proofs or λ -terms as static combinatorial objects, but they can also be seen as programs. In this perspective, unexpected computational consequences of non-commutativity in the λ -calculus have recently been uncovered in an automata-theoretic setting [NP20].

Finally, let us mention Abrusci and Ruet’s logic [AR99, Rue00] where commutative and non-commutative versions of the connectives \otimes and \wp coexist.

Proof nets vs denotational semantics. Pomset logic comes from trying to extract a syntactic correctness criterion from the coherence space semantics: the interpretation of a pre-proof net in coherence spaces can be defined by means of so-called experiments, and we want the result of the experiments to be a valid member of the semantics. (To be precise, we want the set of points obtained by experiments to form a clique.) For MLL+mix proof nets, Retoré showed that this condition is equivalent to correctness [Ret97b], and the correctness criterion for pomset proof nets was designed to extend this correspondence (this is discussed in [Ret97a]).

Pagani has applied a similar methodology to MELL (Multiplicative-Exponential Linear Logic) pre-proof nets: he shows in [Pag06] that the validity of coherence space experiments — using a certain “non-uniform” interpretation of the exponentials — is equivalent to a certain graph-theoretical condition, *visible acyclicity*, which is weaker than the usual correctness criterion for MELL+mix. This is later extended to differential interaction nets in [Pag12]; since coherence spaces are not a semantics of differential linear logic, the result of [Pag12] is formulated with respect to Ehrhard’s finiteness spaces instead.

A similarity between correctness for pomset proof nets and visible acyclicity is that both involve *directed* edges and cycles. Thanks to this, it is straightforward to show that visible acyclicity is **coNP**-hard, by adapting the proof for pomset logic; however we do not know whether, conversely, it is in **coNP**.

Let us also mention Tranquili’s *hypercorrectness* criterion for MALL (Multiplicative-Additive) pre-proof nets, coming from their semantics in hypercoherences [Tra08]. Here again the condition obtained is weaker than the usual correctness criterion — so there are hypercorrect MALL pre-proof nets that are not sequentializable. (Coherence spaces would allow even more non-sequentializable pre-proof nets, for instance a version of Berry’s famous “Gustave function”).

Extensions of BV. Given that BV and pomset logic are “multiplicative” logics, it is natural to make extensions with other primitives, like the additives and exponentials of linear logic, or other modalities or quantifiers.

²⁷This is entirely unrelated to the so-called “cyclic proofs” (also known as “circular proofs”) for logics with fixed points or (co)inductive definitions, a trendy research topic at the time of writing.

This has indeed been done, but so far only for **BV**. The first such extension was adding the exponential of linear logic to **BV**, leading to the logic **NEL**, which is studied in the fourth and fifth paper of this series [SG11, GS11].

In [Rov16], Roversi adds a self-dual binder to **BV**, in order to establish a correspondence to the linear λ -calculus, in the spirit of the *formulas-as-types* paradigm.

The next natural extension was adding the additives, leading to the logic **MAV** [Hor15], which has then been extended by nominal quantifiers (and standard first-order quantifiers) [HTAC19, HT19] in order to simulate private names in process algebras, as for example the π -calculus. This is in the line of research by Bruscoli [Bru02] who used **BV** to simulate reductions in **CCS**, following the *formulas-as-processes* paradigm.

Beyond formulas. The formulas-as-processes paradigm has recently motivated another line of research. The restrictions on digraphs, that define dicographs which correspond to formulas (see Definitions 2.26, 2.27 and Theorem 2.28) and that therefore make proof theory possible in the first place, are also an obstacle to the formulas-as-processes paradigm because there are processes that do contain the forbidden configurations in (2.2) in Definition 2.25, and do therefore not correspond to formulas. This suggests to define a proof system directly on the graphs instead of the formulas, and use the modular decomposition tree instead of the formula tree. This idea (first briefly mentioned in [NS18]) has been explored in [AHS20b, AHS20a] and [CDW20] for undirected graphs and in [AHMS22] for digraphs. It turns out that if we drop the cograph/dicograph condition, there is a much larger space of possible proof systems, that still waits to be explored.

6.2. Open problems. For a long time, it was believed that there was a canonical extension of **MLL**+**mix** with the connective \triangleleft that had both a deductive proof system (**BV**) and proof nets with a simple correctness criterion (pomset logic). Now that we have refuted this, several questions arise:

- We might want to design truly well-behaved deductive proofs for pomset logic — given the obstructions that we have seen in this paper, this looks challenging. Slavnov’s sequent calculus is a start, but it is not clear for us whether a cut-elimination procedure can be defined directly on it without going through a translation into proof nets. And even without insisting on the proof system being deductive, the requirement of tractable proof checking rules out proof nets by themselves (Remark 4.14).
- Our results might also be interpreted as suggesting that of the two logics, **BV** was the “right” one all along. Then it would be desirable to have a system of proof nets for **BV**. Perhaps it suffices to extend the correctness criterion of pomset logic so that it excludes more pre-proofs. If that were the case, then the problem of “**BV**-correctness” of pre-proof nets would be equivalent to the **BV**-provability problem for balanced formulas. This also raises the question of the complexity of the latter: **NP**-completeness would rule out **coNP** criteria of the sort “there does not exist some bad structure (e.g. some kind of cycle) in the pre-proof net”. Alternatively, the right notion of proof net could involve not just the formula tree and axiom linking, but some extra structure too (maybe an order on the axiom linking?); there are some precedents for this in the theory of **MLL** proof nets with units (with many variants, recapitulated in [Hug12, Table 1]).
- More generally, now that we have two logics that (i) are built from the connectives $\wp, \triangleleft, \otimes$, (ii) are conservative over **MLL**+**mix**, and (iii) admit cut elimination, the question

arises whether these are the only two or whether there is a hierarchy of such logics with increasing proof complexity.

During the research for this paper, another interesting question arose. We conjecture the following generalization of the construction of the formula in Section 3: given a balanced tautology of classical logic, one can always “make the axiom links directed” in some way (cf. Remark 3.15) to get a provable formula in pomset logic.

Acknowledgments. We would like to thank Christian Retoré for instructive discussions.

REFERENCES

- [AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity – A Modern Approach*. Cambridge University Press, 2009. URL: <https://theory.cs.princeton.edu/complexity/>.
- [Abr07] Samson Abramsky. Temperley–Lieb Algebra: From Knot Theory to Logic and Computation via Quantum Mechanics. In Goong Chen, Louis Kauffman, and Samuel Lomonaco, editors, *Mathematics of Quantum Computation and Quantum Technology*, volume 20074453, pages 515–558. Chapman and Hall/CRC, September 2007. doi:10.1201/9781584889007.ch15.
- [AHMS22] Matteo Acclavio, Ross Horne, Sjouke Mauw, and Lutz Straßburger. A Graphical Proof Theory of Logical Time. In Amy P. Felty, editor, *7th International Conference on Formal Structures for Computation and Deduction (FSCD 2022)*, volume 228 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 22:1–22:25, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.FSCD.2022.22.
- [AHS20a] Matteo Acclavio, Ross Horne, and Lutz Straßburger. An analytic propositional proof system on graphs. *CoRR*, abs/2012.01102, 2020. Submitted journal version of a LICS’20 paper. arXiv:2012.01102.
- [AHS20b] Matteo Acclavio, Ross Horne, and Lutz Straßburger. Logic beyond formulas: A proof system on graphs. In *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 38–52, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3373718.3394763.
- [AM19] V. Michele Abrusci and Roberto Maieli. Proof nets for multiplicative cyclic linear logic and Lambek calculus. *Mathematical Structures in Computer Science*, pages 1–30, February 2019. doi:10.1017/S0960129518000300.
- [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 *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 6:1–6:16, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.CSL.2020.6.
- [APR05] Jean-Marc Andreoli, Gabriele Pulcini, and Paul Ruet. Permutative logic. In C.-H. Luke Ong, editor, *Computer Science Logic, 19th International Workshop, CSL 2005, 14th Annual Conference of the EACSL, Oxford, UK, August 22-25, 2005, Proceedings*, volume 3634 of *Lecture Notes in Computer Science*, pages 184–199. Springer, 2005. doi:10.1007/11538363_14.
- [AR99] V. Michele Abrusci and Paul Ruet. Non-commutative logic I: the multiplicative fragment. *Annals of Pure and Applied Logic*, 101(1):29–64, 1999. doi:10.1016/S0168-0072(99)00014-7.
- [Bas94] David A. Basin. A term equality problem equivalent to graph isomorphism. *Information Processing Letters*, 51(2):61 – 66, 1994. doi:10.1016/0020-0190(94)00084-0.
- [BdGR97] Denis Bèchet, Philippe de Groote, and Christian Retoré. A complete axiomatisation for the inclusion of series-parallel partial orders. In Hubert Comon, editor, *Rewriting Techniques and Applications, 8th International Conference, RTA-97, Sitges, Spain, June 2-5, 1997, Proceedings*, volume 1232 of *Lecture Notes in Computer Science*, pages 230–240. Springer, 1997. doi:10.1007/3-540-62950-5_74.
- [BG18] Jørgen Bang-Jensen and Gregory Z. Gutin, editors. *Classes of Directed Graphs*. Springer Monographs in Mathematics. Springer, 2018. doi:10.1007/978-3-319-71840-8.

- [BGI⁺14] Richard Blute, Alessio Guglielmi, Ivan T. Ivanov, Prakash Panangaden, and Lutz Straßburger. A logical basis for quantum evolution and entanglement. In Claudia Casadio, Bob Coecke, Michael Moortgat, and Philip Scott, editors, *Categories and Types in Logic, Language, and Physics – Essays Dedicated to Jim Lambek on the Occasion of His 90th Birthday*, volume 8222 of *Lecture Notes in Computer Science*, pages 90–107. Springer, 2014. doi:10.1007/978-3-642-54789-8_6.
- [BPS12] Richard Blute, Prakash Panangaden, and Sergey Slavnov. Deep inference and probabilistic coherence spaces. *Applied Categorical Structures*, 20(3):209–228, 2012. doi:10.1007/s10485-010-9241-0.
- [Bru02] Paola Bruscoli. A purely logical account of sequentiality in proof search. In Peter J. Stuckey, editor, *Logic Programming, 18th International Conference, ICLP 2002, Copenhagen, Denmark, July 29 - August 1, 2002, Proceedings*, volume 2401 of *Lecture Notes in Computer Science*, pages 302–316. Springer, 2002. doi:10.1007/3-540-45619-8_21.
- [BT01] Kai Brännler and Alwen Fernanto Tiu. A local system for classical logic. In Robert Nieuwenhuis and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning, 8th International Conference, LPAR 2001, Havana, Cuba, December 3-7, 2001, Proceedings*, volume 2250 of *Lecture Notes in Computer Science*, pages 347–361. Springer, 2001. doi:10.1007/3-540-45653-8_24.
- [CDW20] Cameron Calk, Anupam Das, and Tim Waring. Beyond formulas-as-cographs: an extension of boolean logic to arbitrary graphs. *CoRR*, abs/2004.12941, 2020. arXiv:2004.12941.
- [CLB81] Derek G. Corneil, H. Lerchs, and L. Stewart Burlingham. Complement reducible graphs. *Discrete Applied Mathematics*, 3(3):163–174, 1981. doi:10.1016/0166-218X(81)90013-5.
- [CP06] Christophe Crespelle and Christophe Paul. Fully dynamic recognition algorithm and certificate for directed cographs. *Discrete Applied Mathematics*, 154(12):1722–1741, 2006. doi:10.1016/j.dam.2006.03.005.
- [CR79] Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44(1):36–50, 1979. doi:10.2307/2273702.
- [DG04] Pietro Di Gianantonio. Structures for Multiplicative Cyclic Linear Logic: Deepness vs Cyclicity. In *Computer Science Logic*, volume 3210, pages 130–144, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. doi:10.1007/978-3-540-30124-0_13.
- [dP14] Valeria de Paiva. Linear logic model of state revisited. *Logic Journal of the IGPL*, 22(5):791–804, 2014. doi:10.1093/jigpal/jzu013.
- [DR89] Vincent Danos and Laurent Regnier. The structure of multiplicatives. *Archive for Mathematical Logic*, 28(3):181–203, 1989. doi:10.1007/BF01622878.
- [FR94] Arnaud Fleury and Christian Retoré. The mix rule. *Mathematical Structures in Computer Science*, 4(2):273–285, 1994. doi:10.1017/S0960129500000451.
- [Gir87] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50(1):1–101, January 1987. doi:10.1016/0304-3975(87)90045-4.
- [Gir89] Jean-Yves Girard. Towards a geometry of interaction. In J. W. Gray and A. Scedrov, editors, *Categories in Computer Science and Logic*, volume 92 of *Contemporary Mathematics*, pages 69–108. American Mathematical Society, Providence, RI, 1989. Proceedings of a Summer Research Conference held June 14–20, 1987. doi:10.1090/conm/092/1003197.
- [GLMM13] Laurent Gourvès, Adria Lyra, Carlos A. Martinhon, and Jérôme Monnot. Complexity of trails, paths and circuits in arc-colored digraphs. *Discrete Applied Mathematics*, 161(6):819–828, April 2013. doi:10.1016/j.dam.2012.10.025.
- [GS01] Alessio Guglielmi and Lutz Straßburger. Non-commutativity and MELL in the Calculus of Structures. In Gerhard Goos, Juris Hartmanis, Jan van Leeuwen, and Laurent Fribourg, editors, *Computer Science Logic*, volume 2142, pages 54–68. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001. doi:10.1007/3-540-44802-0_5.
- [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:10.1017/S096012951100003X.
- [Gug99] Alessio Guglielmi. A calculus of order and interaction, 1999. Obsolete version of [Gug07]. URL: <https://arxiv.org/abs/cs/9910023v1>.

- [Gug07] Alessio Guglielmi. A system of interaction and structure. *ACM Transactions on Computational Logic*, 8(1), January 2007. According to the author, the published version contains errors introduced by the editorial processing; the authoritative version is <https://arxiv.org/abs/cs/9910023>. doi:10.1145/1182613.1182614.
- [Hor15] Ross 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.
- [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 Transactions on Computational Logic*, 20(4):22:1–22:44, 2019. doi:10.1145/3325821.
- [Hug04] Dominic J.D. Hughes. Deep inference proof theory equals categorical proof theory minus coherence. Unfinished draft, 2004. URL: <http://boole.stanford.edu/~dominic/papers/di/di.pdf>.
- [Hug12] Dominic J.D. Hughes. Simple free star-autonomous categories and full coherence. *Journal of Pure and Applied Algebra*, 216(11):2386–2410, 2012. doi:10.1016/j.jpaa.2012.03.020.
- [Kah04] Ozan Kahramanoğulları. System BV without the Equalities for Unit. In *Computer and Information Sciences - ISCIS 2004*, volume 3280, pages 986–995. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004. doi:10.1007/978-3-540-30182-0_99.
- [Kah08] Ozan Kahramanoğulları. System BV is NP-complete. *Annals of Pure and Applied Logic*, 152(1):107–121, March 2008. doi:10.1016/j.apal.2007.11.005.
- [Lam58] Joachim Lambek. The mathematics of sentence structure. *The American Mathematical Monthly*, 65(3):154–170, 1958. URL: <http://www.jstor.org/stable/2310058>.
- [Mel18] Paul-André Melliès. Ribbon Tensorial Logic. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science - LICS '18*, pages 689–698, Oxford, United Kingdom, 2018. ACM Press. doi:10.1145/3209108.3209129.
- [Möh89] Rolf H. Möhring. Computationally tractable classes of ordered sets. In I. Rival, editor, *Algorithms and Order*, pages 105–194. Kluwer Acad. Publ., 1989.
- [MR12] Richard Moot and Christian Retoré. *The Logic of Categorical Grammars*, volume 6850 of *Lecture Notes in Computer Science*. Springer, 2012. doi:10.1007/978-3-642-31555-8.
- [Ngu19] Lê Thành Dũng (Tito) Nguyễn. Constrained path-finding and structure from acyclicity. *CoRR*, abs/1901.07028, 2019. arXiv:1901.07028.
- [Ngu20] Lê Thành Dũng (Tito) Nguyễn. Unique perfect matchings, forbidden transitions and proof nets for linear logic with Mix. *Logical Methods in Computer Science*, Volume 16, Issue 1(27):27:1–27:31, February 2020. doi:10.23638/LMCS-16(1:27)2020.
- [NP20] Lê Thành Dũng (Tito) Nguyễn and Pierre Pradic. Implicit Automata in Typed λ -Calculi I: Aperiodicity in a Non-Commutative Logic. In Artur Czumaj, Anuj Dawar, and Emanuela Merelli, editors, *47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*, volume 168 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 135:1–135:20, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2020.135.
- [NS18] Lê Thành Dũng (Tito) Nguyễn and Thomas Seiller. Coherent interaction graphs. In Thomas Ehrhard, Maribel Fernández, Valeria de Paiva, and Lorenzo Tortora de Falco, editors, *Proceedings Joint International Workshop on Linearity & Trends in Linear Logic and Applications, Linearity-TLLA@FLoC 2018, Oxford, UK, 7-8 July 2018*, volume 292 of *EPTCS*, pages 104–117, 2018. doi:10.4204/EPTCS.292.6.
- [NS22] Lê Thành Dũng (Tito) Nguyễn 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. doi:10.4230/LIPIcs.CSL.2022.32.
- [OP99] Peter W. O’Hearn and David J. Pym. The logic of bunched implications. *Bulletin of Symbolic Logic*, 5(2):215–244, 1999. doi:10.2307/421090.
- [Pag06] Michele Pagani. Acyclicity and coherence in multiplicative exponential linear logic. In Zoltán Ésik, editor, *Computer Science Logic, 20th International Workshop, CSL 2006, 15th Annual*

- Conference of the EACSL, Szeged, Hungary, September 25-29, 2006, Proceedings*, volume 4207 of *Lecture Notes in Computer Science*, pages 531–545. Springer, 2006. doi:10.1007/11874683_35.
- [Pag12] Michele Pagani. Visible acyclic differential nets, Part I: Semantics. *Annals of Pure and Applied Logic*, 163(3):238–265, March 2012. doi:10.1016/j.apal.2011.09.001.
- [Red93] Uday S. Reddy. A linear logic model of state, 1993. Unpublished note. URL: <https://www.cs.bham.ac.uk/~udr/papers/state.full.pdf>.
- [Ret93] Christian Retoré. *Réseaux et séquents ordonnés*. PhD thesis, Université Paris VII, February 1993. URL: <https://tel.archives-ouvertes.fr/tel-00585634>.
- [Ret97a] Christian Retoré. Pomset logic: A non-commutative extension of classical linear logic. In *Typed Lambda Calculi and Applications*, volume 1210, pages 300–318. Springer Berlin Heidelberg, Berlin, Heidelberg, 1997. doi:10.1007/3-540-62688-3_43.
- [Ret97b] Christian Retoré. A semantic characterisation of the correctness of a proof net. *Mathematical Structures in Computer Science*, 7(5):445–452, October 1997. doi:10.1017/S096012959700234X.
- [Ret99a] Christian Retoré. Handsome proof-nets: R&B-graphs, perfect matchings and series-parallel graphs. Research Report 3652, INRIA, March 1999. URL: <https://hal.inria.fr/inria-00073020>.
- [Ret99b] Christian Retoré. Pomset Logic as a Calculus of Directed Cographs. Research Report 3714, INRIA, June 1999. URL: <https://hal.inria.fr/inria-00072953>.
- [Ret03] Christian Retoré. Handsome proof-nets: perfect matchings and cographs. *Theoretical Computer Science*, 294(3):473–488, February 2003. doi:10.1016/S0304-3975(01)00175-X.
- [Ret21] Christian Retoré. Pomset Logic: The other approach to noncommutativity in logic. In Claudia Casadio and Philip J. Scott, editors, *Joachim Lambek: the interplay of mathematics, logic, and linguistics*, Outstanding Contributions to Logic, chapter 9, pages 299–346. Springer, 2021. doi:10.1007/978-3-030-66545-6_9.
- [Rov16] Luca Roversi. A deep inference system with a self-dual binder which is complete for linear lambda calculus. *J. Log. Comput.*, 26(2):677–698, 2016. doi:10.1093/logcom/exu033.
- [Rue00] Paul Ruet. Non-commutative logic II: sequent calculus and phase semantics. *Mathematical Structures in Computer Science*, 10(2):277–312, April 2000. doi:10.1017/S0960129599003084.
- [SG11] Lutz Straßburger and Alessio Guglielmi. A system of interaction and structure IV: the exponentials and decomposition. *ACM Transactions on Computational Logic*, 12(4):23:1–23:39, 2011. doi:10.1145/1970398.1970399.
- [SK22] Will Simmons and Aleks Kissinger. Higher-Order Causal Theories Are Models of BV-Logic. In Stefan Szeider, Robert Ganian, and Alexandra Silva, editors, *47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022)*, volume 241 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 80:1–80:14, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.MFCS.2022.80.
- [Sla19] Sergey Slavnov. On noncommutative extensions of linear logic. *Logical Methods in Computer Science*, Volume 15, Issue 3, September 2019. doi:10.23638/LMCS-15(3:30)2019.
- [Str03a] Lutz Straßburger. System NEL is undecidable. *Electronic Notes in Theoretical Computer Science*, 84:166–177, 2003. doi:10.1016/S1571-0661(04)80853-3.
- [Str03b] Lutz Straßburger. *Linear Logic and Noncommutativity in the Calculus of Structures*. PhD Thesis, Technische Universität Dresden, 2003. URL: <https://www.lix.polytechnique.fr/~lutz/papers/dissvonlutz.pdf>.
- [Str12] Lutz Straßburger. Extension without cut. *Annals of Pure and Applied Logic*, 163(12):1995–2007, 2012. doi:10.1016/j.apal.2012.07.004.
- [Tiu06] Alwen Tiu. A System of Interaction and Structure II: The Need for Deep Inference. *Logical Methods in Computer Science*, 2(2):4, April 2006. doi:10.2168/LMCS-2(2:4)2006.
- [Tra08] Paolo Tranquilli. A Characterization of Hypercoherent Semantic Correctness in Multiplicative Additive Linear Logic. In Michael Kaminski and Simone Martini, editors, *Computer Science Logic*, volume 5213, pages 246–261. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. doi:10.1007/978-3-540-87531-4_19.
- [TS19] Andrea Aler Tubella and Lutz Strassburger. Introduction to Deep Inference. Lecture notes, August 2019. URL: <https://hal.inria.fr/hal-02390267>.
- [VTL82] Jacobo Valdes, Robert Endre Tarjan, and Eugene L. Lawler. The recognition of series parallel digraphs. *SIAM Journal on Computing*, 11(2):298–313, 1982. doi:10.1137/0211023.

- [Wra76] Celia Wrathall. Complete sets and the polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):23–33, October 1976. doi:10.1016/0304-3975(76)90062-1.
- [Yet90] David N. Yetter. Quantales and (noncommutative) linear logic. *The Journal of Symbolic Logic*, 55(1):41–64, March 1990. doi:10.2307/2274953.
- [ZG15] Noam Zeilberger and Alain Giorgetti. A correspondence between rooted planar maps and normal planar lambda terms. *Log. Methods Comput. Sci.*, 11(3), 2015. doi:10.2168/LMCS-11(3:22)2015.