Submitted to MFPS 2023

The exponential logic of sequentialization

Aurore Alcolei[a,1] Luc Pellissier[b,2] Alexis Saurin[c,3]

[a] *Université Paris Est Creteil, LACL, F-94010 Créteil, France*

[b] *Université Paris Est Creteil, LACL, F-94010 Créteil, France*

[c] *IRIF, CNRS, Université Paris Cité & INRIA, Paris, France*

Abstract

Linear logic has provided new perspectives on proof-theory, denotational semantics and the study of programming languages. One of its main successes are proof-nets, canonical representations of proofs that lie at the intersection between logic and graph theory. In the case of the minimalist proof-system of multiplicative linear logic without units (MLL), these two aspects are completely fused: proof-nets for this system are graphs satisfying a correctness criterion that can be fully expressed in the language of graphs.

For more expressive logical systems (containing logical constants, quantifiers and exponential modalities), this is not completely the case. The purely graphical approach of proof-nets deprives them of any sequential structure that is crucial to represent the order in which arguments are presented, which is necessary for these extensions. Rebuilding this order of presentation — sequentializing the graph — is thus a requirement for a graph to be logical. Presentations and study of the artifacts ensuring that sequentialization can be done, such as boxes or jumps, are an integral part of researches on linear logic.

Jumps, extensively studied by Faggian and di Giamberardino, can express intermediate degrees of sequentialization between a sequent calculus proof and a fully desequentialized proof-net. We propose to analyze the logical strength of jumps by internalizing them in an extention of MLL where axioms on a specific formula, the jumping formula, introduce constrains on the possible sequentializations. The jumping formula needs to be treated non-linearly, which we do either axiomatically, or by embedding it in a very controlled fragment of multiplicative-exponential linear logic, uncovering the exponential logic of sequentialization.

Keywords: jumps, linear logic, proof theory, proof nets, sequentialization

1 Introduction

Proof theory is concerned with proof systems, specifying how proofs are structured, and in turn, which formulæ are provable. Different proof systems have different properties, such as soundness (the fact that all provable formulæ are also valid — for a given semantics of formulæ), completeness (the fact that all valid formulæ are also provable — again, for a given semantics of formulæ), canonicity (two proofs differ if and only if they have different interpretations — for a given semantics of proofs), ...

Broadly speaking, Linear logic [9] has two main kinds of proof systems:

Sequent calculus Proofs are trees whose nodes are labelled by inference rules transforming sequents of formulæ. This representation – even though it is both historically decisive in the development of proof-theory since the work of Gentzen by emphasizing the role of the cut inference as well as for

[1] aurore.alcolei@ens-lyon.org

[2] luc.pellissier@u-pec.fr

[3] alexis.saurin@irif.fr

reductive logics (aka. proof-construction) by suggesting a natural notion of proof-goal – contains lot of unnecessary information (such as the specific order of application of some unrelated rules): it is far from being canonical. The loss of confluence in sequent calculus witnesses this loss of canonicity.

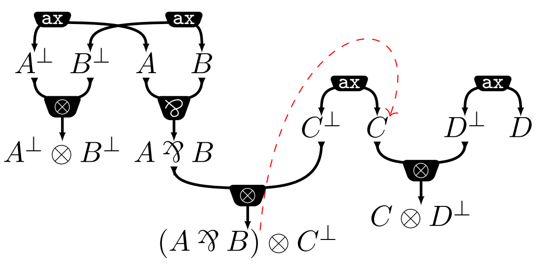
Proof-nets are graphs whose nodes relate formulæ to one another. A proof-net contains much less information than a sequent calculus proof with respect to the order of application of inference rules. This allows them to be canonical, for some fragments of linear logic at least. In particular, they are canonical for multiplicative linear logic without units (simply referred to as MLL in the following). As a by-product, confluence is recovered in MLL proof-nets.

Apart from the contrast on canonicity issue, a major difference between sequent proofs and proof-nets follows from the tree *vs.* graph structure and concerns logical correctness of the proof-object: sequent proofs being inductively defined proof objects, their logical correctness is naturally defined in a *local* way: soundness of the conclusion follows from the soundness of each premise and one only considers derivation trees that is built applying inferences in a correct way. On the other hand, proof-nets being graphs, there is no such thing as a root, in general, and logical correctness therefore becomes a *global* property of the graph. It would be ad hoc to restrict solely to those graphs which are logically correct and one therefore consider two levels of proof objects: **proof structures**, which are graphs with some typing constraints on the edges and vertices; and **proof-nets**, which are logically correct proof structures.

A way to introduce contexts (other than boxes [9]) are *jumps*, a recurrent feature in the literature on proof nets [10].

They are purely geometrical (i.e. untyped) edges adding extra connection between links of a proof structure. Jumps were first introduced by Girard in [10] to add more sequentiality in proof nets with quantifiers, typically preventing sequentialization that would not match the variable-witness dependencies between $\forall$ and $\exists$ quantifier. Jumps are also used as lighter feature than boxes to handle the sequentiality induced by sequent rules outside of MLL, typically the unit rules [2,11,13], additive rules [11,12] and exponentials for λ-nets [1].

Fig. 1: A graphical jump



On another line of work, Faggian and her collaborators developped a setting in which jumps are taken seriously as edges in a more general graph than just the mere syntactic tree of a formula. First in the context of *L-nets* [3,4,8], a parallel syntax for Ludics designs, then in the context of *J-proof nets* for HS^+ (a polarized and hypersequentialized sequent calculus) and MLL [6,7], their setting allows in particular to see sequent calculus proofs as proof-nets saturated with sequential edges, embedding both proof-nets and sequent calculus proofs in a same universe of partially sequentialized proof-structures, each of them living at one extremity on a de·sequentialization spectrum.

The figure above depicts a proof net with a jump (the red dashed arrow) from (the formula occurrence) $(A \parr B) \otimes C^\perp$ to (the formula occurrence) C. As a consequence, when sequentializing such a net, the $\otimes$ rule introducing $(A \parr B) \otimes C^\perp$ will always be placed above the $\otimes$ rule using C to introduce $C \otimes D^\perp$. Jumps are thus a way to *constrain* the set of sequentializations that are derivable from a proof net. As additional features in the usual proof net theory, usual correctness criteria on proof nets must be adapted to the setting. In this paper, we show how those graphical edges (and their dynamics) can in fact be internalised directly in linear logic, that is encoded inside (a variant of) the logic, in particular retrieving their correctness criteria through the usual ones.

Contributions and organization of the paper.

In this paper, we show that graphical jumps as presented above can be internalised in the logic under study at the only cost of adding a special atom $\mathbf{J}$ that enjoys, as well as its dual, the property of a monoid and behaves as an exponential under cut elimination.

To keep this paper light and readable, we will focus our internalisation of jumps to MLL. In Section 4, we will recast the basic definitions on MLL sequent proofs and proof-nets. In Section 3 we start introducing our encoding of jumps as axioms in a cut-free setting. In particular, we will define $\mathsf{MLL}_\mathbf{J}$ a logic which is basically MLL plus a special atom $\mathbf{J}$ that enjoys operadic logical rules. We will prove that our encoding is correct and that our jumps correspond exactly to the the one described in [7]. In Section **??** we will discuss the encoding of jumps in the dynamic setting of proofs with cuts and cut elimination, enriching $\mathsf{MLL}_{\mathbf{J}}$ with the appropriate exponential dynamics for $\mathbf{J}$. This will make explicit the dynamics of jumps sketched in [7] and treated more exhaustively in [5]. Finally, we will conclude in Section 5 by showing that the jump atom $\mathbf{J}$ can be implemented effectively in the exponential fragment of MELL + Mix.

2 MLL proofs: sequent-calculus, proof-nets and sequentialization

Sequent calculus proofs as (constrained) proof structures

A formula F is a *formula occurrence* in π if it is the active conclusion of a rule in π. For F a formula occurrence in π, we define its *life-time* as sub-branch from its creation to its use.

For two rule occurrences r and r' in a proof π, we say that r is *above* (or *occurs before*) r' if r and r' belong to the same branch in the proof, and r' appears before r starting from the root. If r produces a formula F this is equivalent to say that there exists F' an active formula of r' such that F is created before F' is used. We write $F \prec F'$.

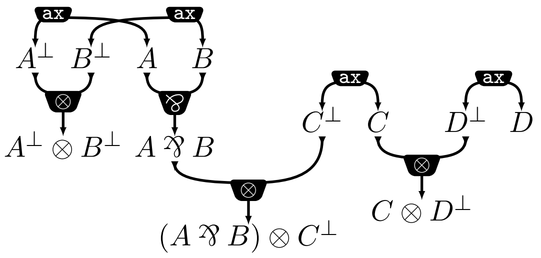
Every sequent proof π can be desequentialized into a proof structure R_π, defined inductively. For a proof structure R, we note $\mathsf{seq}(R)$ its set of sequentializations, that is, the set of sequent proofs π such that $R_\pi = R$. Desequentialization is preserved by cut elimination and, conversely, if $\pi \in \mathsf{seq}(R)$ and $R \to R'$, then there exists $\pi' \in \mathsf{seq}(R')$ such that $\pi \to^+ \pi'$. Note, however, that in general proofs in $\mathsf{seq}(R')$ do not correspond to the reduct of some proof in $\mathsf{seq}(R)$.

If π is a sequentialization of R then there is a one-to-one correspondence between the rules in π and the links in R (this correspondence is defined during the inductive procedure of desequentialization). By extension every edge in R can be associated with a formula occurrence in π throughout its life-time. In the following we will often make use of this correspondence, keeping the conversion from proof to net implicit.

Given a proof structure R, its set of sequentializations may be empty, in that case we say that R is *incorrect*. On the contrary, if $\mathsf{seq}(R)$ is non empty, we say that R is *correct* or that it is *a proof net*, meaning that it actually corresponds to a canonical representation of proofs.

A proof net can have several sequentializations, for example:

Example 2.1 Consider the (correct) proof-structure R:



It has two different sequentializations:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash A^\perp, A} \quad \dfrac{}{\vdash B^\perp, B}}{\vdash A^\perp \otimes B^\perp, A, B}\otimes}{\vdash A^\perp \otimes B^\perp, A \parr B}\parr \quad \dfrac{}{\vdash C^\perp, C}}{\vdash A^\perp \otimes B^\perp, (A \parr B) \otimes C^\perp, C}\otimes \quad \dfrac{}{\vdash D^\perp, D}}{\vdash A^\perp \otimes B^\perp, (A \parr B) \otimes C^\perp, C \otimes D^\perp, D}\otimes \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{\vdash A^\perp, A} \quad \dfrac{}{\vdash B^\perp, B}}{\vdash A^\perp \otimes B^\perp, A, B}\otimes}{\vdash A^\perp \otimes B^\perp, A \parr B}\parr \quad \dfrac{\dfrac{}{\vdash C^\perp, C} \quad \dfrac{}{\vdash D^\perp, D}}{\vdash C^\perp, C \otimes D^\perp, D}\otimes}{\vdash A^\perp \otimes B^\perp, (A \parr B) \otimes C^\perp, C \otimes D^\perp, D}\otimes$$

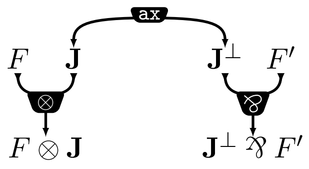
On the first one, A is above D, not in the second.

Graphical jumps as depicted in Figure 1 are used to restrict the possible set of sequentializations of a net: if there is a jump from F to F' in R, then we expect F to appear above F' in any sequentialization of R. In the above example, only the left sequentialization would be possible.

3 Jumps as axioms

The graphical jump presented above can be encoded in cut-free MLL proof structures using the axiom link of fresh atomic formulas $\mathbf{J}$ to create a synchronisation point.

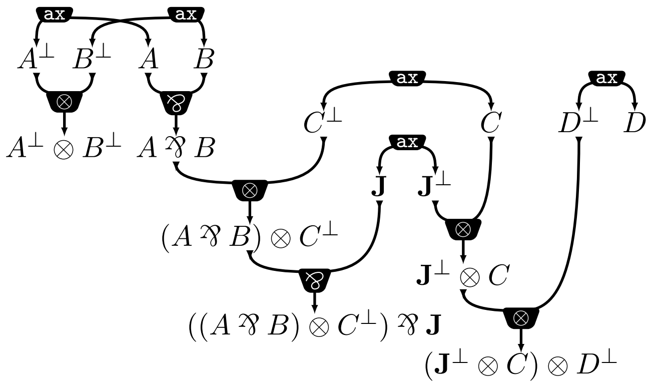
Definition 3.1 Let F and F' be two formulæ. A *jump* from F to F' is the gadget:



Let R be a proof-structure, and F, F' be two formula occurrences in R.

We define $\mathbf{J}_{F\to F'}(R)$ the *proof structure with a jump* from F to F' by grafting the jump from F to F' where F and F' are, and making all the types coherent.

Let us illustrate the previous definition. Consider the (correct) proof-structure R in Example 2.1, we can add a jump from $F = (A \parr B) \otimes C^\perp$ to C, resulting in the proof-structure $\mathbf{J}_{F\to C}(R)$:


ax
ax
$A^\perp$
$B^\perp$
A
B
$A^\perp \otimes B^\perp$
$A \parr B$
ax
$C^\perp$
C
ax
$D^\perp$
D
ax
$\mathbf{J}$
$\mathbf{J}^\perp$
$(A \parr B) \otimes C^\perp$
$\mathbf{J}^\perp \otimes C$
$((A \parr B) \otimes C^\perp) \parr \mathbf{J}$
$(\mathbf{J}^\perp \otimes C) \otimes D^\perp$

This proof-structure is much more rigid than R. It also has only two sequentializations:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdash A^\perp, A \qquad \vdash B^\perp, B}{\vdash A^\perp \otimes B^\perp, A, B}\otimes}{\vdash A^\perp \otimes B^\perp, A \parr B}\parr \qquad \vdash C^\perp, C}{\vdash A^\perp \otimes B^\perp, (A \parr B) \otimes C^\perp, C}\otimes \qquad \vdash \mathbf{J}^\perp, \mathbf{J}}{\vdash A^\perp \otimes B^\perp, (A \parr B) \otimes C^\perp, \mathbf{J}, \mathbf{J}^\perp \otimes C}\otimes}{\vdash A^\perp \otimes B^\perp, (A \parr B) \otimes C^\perp \parr \mathbf{J}, \mathbf{J}^\perp \otimes C}\parr \qquad \vdash D^\perp, D}{\vdash A^\perp \otimes B^\perp, (A \parr B) \otimes C^\perp \parr \mathbf{J}, (\mathbf{J}^\perp \otimes C) \otimes D^\perp, D}\otimes$$

and the one where the $\parr$ rule shown in green is commuted below the $\otimes$ rule. They differ in a much lighter way than the two sequentializations of R: indeed, forgetting every occurrence of $\mathbf{J}$ in both of these two sequent calculus proofs yields only one of the two sequentializations of the original proof-net. The presence of the jump from $F = (A \parr B) \otimes C^\perp$ to C forces F to be above C in every sequentialization of $\mathbf{J}_{F\to C}(R)$. Let us turn this example into a general theorem.

Theorem 3.2 (correctness) *Let R be a proof-structure, and F and F' two formula occurrences, then $\mathbf{J}_{F\to F'}(R)$ is correct if and only if there exists a sequentialization of R such that F is above F'.*

4 General case

This encoding of jumps have to be complexified in order to allow for multiple jumps and cuts. To this end, we define $\mathsf{MLL}_\mathbf{J}$ which can be seen as an extension of MLL with a special atom $\mathbf{J}$ enjoying contractions as

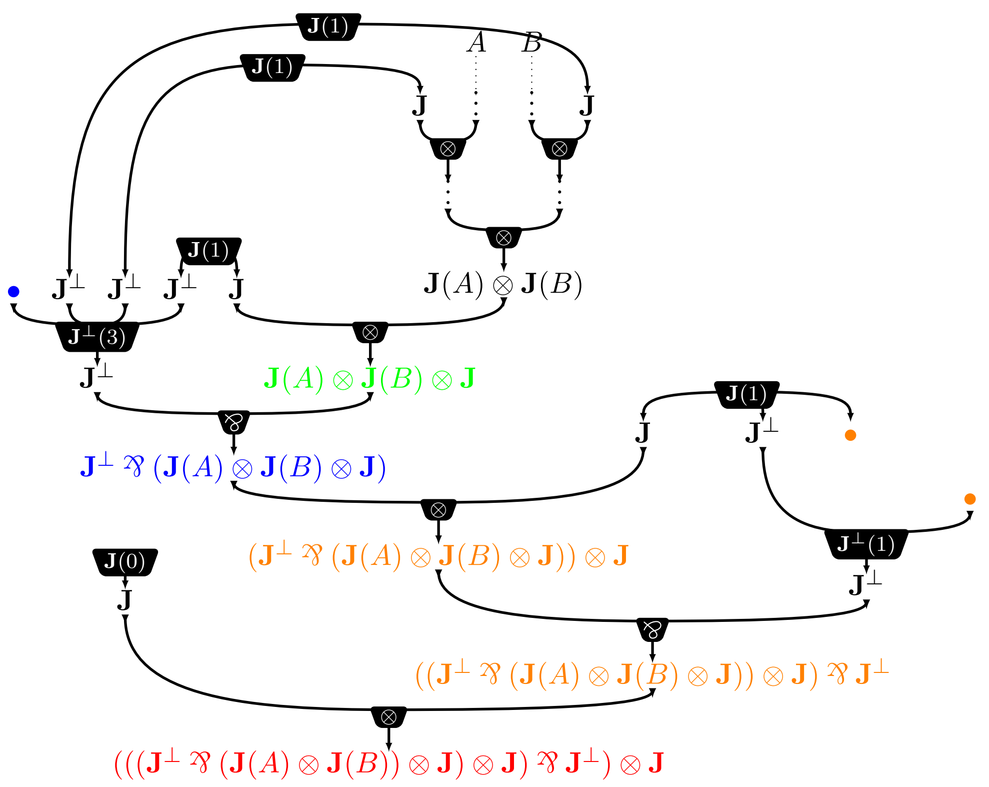


Fig. 2. Gadget for a positive binary connective $A \otimes B$

depicted in Figure 3. We then replace each connective and formula by a more complex gadget. Consider for instance a positive connective $A \otimes B$, we replace it by the gadget in Figure 2

- As before, Orange links define the *out- and in-synchronisation points* of an edge F in R.
- Blue and Green links are *propagation* points, they will allow the incoming jumps of the premises of a link to inherit from the outgoing jumps of the link during cut elimination (or the outgoing jumps of the conclusion of an axiom link to be transfered to the positive atom after reduction). In particular, Blue links are connected to the out-synchronisation point of the premises of the link being replaced.
- Finally, Red links are *absorption* points, they will erase, the incoming jump of the premises of a cut after its elimination.

Given two formula occurrences F and F' in R, a *jump from F to F'* is now encoded in R_J by connecting the out-synchronisation point of F to the in-synchronisation point of F', *and*, if F' is the premise of a link in R, to the Blue propagation point of that link.

Let $\mathcal{L}$ be a set of pairs of formula occurrences in R, we note $\mathbf{J}_{\mathcal{L}}(R)$ the $\mathsf{MLL}_{\mathbf{J}}$-proof structure corresponding to $\mathbf{J}(R)$ with all the jumps described in $\mathcal{L}$.

$$\begin{array}{l}
\dfrac{}{\vdash J, \underbrace{J^{\perp},\dots,J^{\perp}}_{n}}\,\mathrm{Ax}(n) \\[2ex]
\dfrac{}{\vdash \Gamma, \underbrace{J^{\perp},\dots,J^{\perp}}_{n}}\,\mathrm{ctr}(n)
\end{array}$$

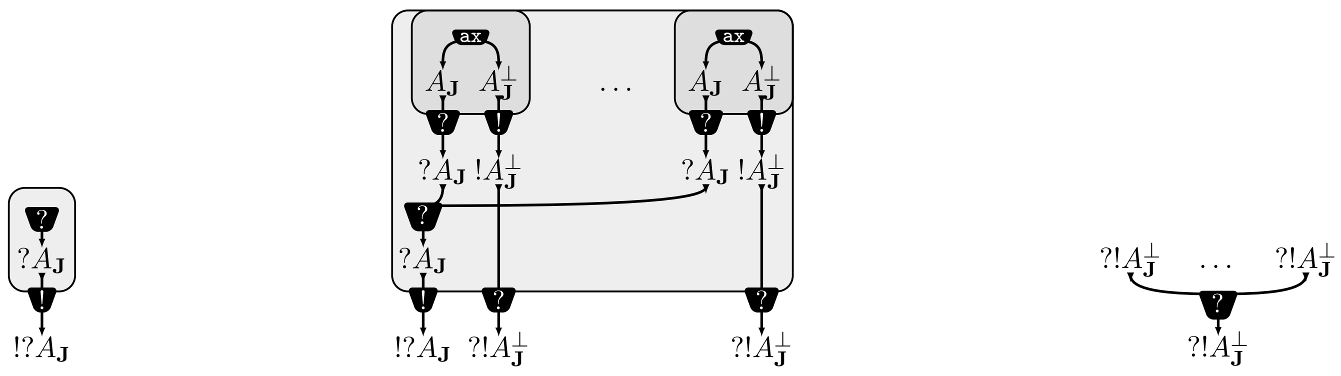
Fig. 3: Rules for $\mathbf{J}$

5 Implementation

In this section, in order to finish our logical internalization of jumps, we eventually show how jumps, as defined in Figure 3, can be implemented in the exponential fragment of $\mathsf{MELL} + \mathsf{Mix}$.

As mentioned in Section 4, we need $\mathbf{J}$ and $\mathbf{J}^{\perp}$ to be able to contract. This can be achieved using the contraction rule of the ?-exponential in MELL, so our encoding of $\mathbf{J}$ and $\mathbf{J}^{\perp}$ must involve this exponential modality. $\mathbf{J}$ only needs the ability to contract at the level of axioms while $\mathbf{J}^{\perp}$ is free to contract everywhere, furthermore, $\mathbf{J}$ and $\mathbf{J}^{\perp}$ must be dual from each other, as such, we set:

$$\mathbf{J} = !?A_{\mathbf{J}} \qquad\qquad \mathbf{J}^{\perp} = ?!A_{\mathbf{J}}^{\perp}$$



(a) $\mathbf{J}(0)$ (b) $\mathbf{J}(n), n > 0$ (c) $\mathbf{J}^{\perp}(n), n > 0$

Fig. 4. Encoding of the combinators on $\mathbf{J}$ and $\mathbf{J}^{\perp}$

where $A_{\mathbf{J}}$ is a special atom that does not appear in regular MLL formulae.

In Figure 4 we show how the static properties of jumps (that are $\mathbf{J}(n \geq 0)$ and $\mathbf{J}^{\perp}(m > 0)$) can be derived via this encoding. The dynamics properties of jumps can then be derived from the usual dynamics of exponential in MELL.

References

[1] Accattoli, B. and S. Guerrini, *Jumping boxes*, in: *CSL*, Lecture Notes in Computer Science **5771** (2009), pp. 55–70.

[2] Chouquet, J. and L. V. Auclair, *An application of parallel cut elimination in multiplicative linear logic to the Taylor expansion of proof nets*, Logical Methods in Computer Science **Volume 17, Issue 4** (2021).
URL https://lmcs.episciences.org/8871

[3] Curien, P. and C. Faggian, *L-nets, strategies and proof-nets*, in: C. L. Ong, editor, *Computer Science Logic, 19th International Workshop, CSL 2005, 14th Annual Conference of the EACSL, Oxford, UK, August 22-25, 2005, Proceedings*, Lecture Notes in Computer Science **3634** (2005), pp. 167–183.
URL https://doi.org/10.1007/11538363_13

[4] Curien, P. and C. Faggian, *An approach to innocent strategies as graphs*, Inf. Comput. **214** (2012), pp. 119–155.
URL https://doi.org/10.1016/j.ic.2011.12.006

[5] Di Giamberardino, P., *Jump from parallel to sequential proofs: exponentials*, Mathematical Structures in Computer Science **28** (2018), p. 1204–1252.

[6] Di Giamberardino, P. and C. Faggian, *Jump from parallel to sequential proofs: Multiplicatives*, in: Z. Ésik, editor, *Computer Science Logic* (2006), pp. 319–333.

[7] Di Giamberardino, P. and C. Faggian, *Proof nets sequentialisation in multiplicative linear logic*, Annals of Pure and Applied Logic **155** (2008), pp. 173–182.
URL https://www.sciencedirect.com/science/article/pii/S0168007208000511

[8] Faggian, C. and F. Maurel, *Ludics nets, a game model of concurrent interaction*, in: *20th IEEE Symposium on Logic in Computer Science (LICS 2005), 26-29 June 2005, Chicago, IL, USA, Proceedings* (2005), pp. 376–385.
URL https://doi.org/10.1109/LICS.2005.25

[9] Girard, J., *Linear logic*, Theor. Comput. Sci. **50** (1987), pp. 1–102.

[10] Girard, J.-Y., *Quantifiers in linear logic II*, in: *Nuovi problemi della logica e della filosofia delle scienze*, CLUEB **II**, 1991, pp. 79–89.

[11] Girard, J.-Y., *Proof-nets: The parallel syntax for proof-theory*, in: *Logic and Algebra* (1996), pp. 97–124.

[12] Hughes, D. J. D. and R. J. van Glabbeek, *Proof nets for unit-free multiplicative-additive linear logic*, ACM Trans. Comput. Log. **6** (2005), pp. 784–842.

[13] Laurent, O., *Polarized proof-nets: Proof-nets for LC*, in: *TLCA*, Lecture Notes in Computer Science **1581** (1999), pp. 213–227.