

Geometry of grammar

Exercises in Lambek style

Mario Fadda

Universitat Politècnica de Catalunya
Jordi Girona Salgado 1-3
E - 08034 Barcelona

Director de la tesi: prof. Glyn Morrill
Doctorat en Intel·ligència Artificial
Departament de Llenguatges i Sistemes Informàtics

Barcelona, juny-juliol 2010

Tesi presentada per obtenir el títol de Doctor
per la Universitat Politècnica de Catalunya

Contents

Abstract	xi
Acknowledgments	xiii
Introduction	1
I Beyond monomodality	13
1 Preliminaries	15
1.1 Structuring information	19
1.1.1 Trees	19
1.1.2 Seaweeds	24
1.2 Checking information	29
1.2.1 Defining proof structures	29
1.2.2 Working with proof structures	38
2 The Diamond of Basic Calculi	49
2.1 Presentation of the calculi	54
2.1.1 Linear Logic φ - LL	54
2.1.2 Lambek Calculus φ - LC	60

2.1.3	Embedding φ - LC into φ - LL	64
2.2	Proof net theory	66
2.2.1	Proof net theory for φ - LL	66
2.2.2	Proof net theory for φ - LC	74
2.2.3	Planar proof nets	76
3	Combining and Constraining Basic Calculi	81
3.1	Endomodal Calculi	83
3.1.1	Linear Logic Φ - LL	83
3.1.2	Lambek Calculi Φ - LC	88
3.2	Bracketed Calculi	93
3.2.1	Linear Logic Φ - LLb	93
3.2.2	Lambek Calculus Φ - LCb	94
3.2.3	Mimicking brackets with two extra symbols	98
3.2.4	Mimicking brackets with one extra symbol	103
II	Beyond multimodality	111
4	Discontinuous Lambek Calculus	113
4.1	The Hypersequent Calculus BDLC	117
4.2	The Hypersequent Calculus HC_e^ω	123
4.2.1	Types and hypersequents	125
4.2.2	Rules of the calculus	128
4.3	Proof nets for HC_e^ω	129
4.3.1	Proof structures and correctness criteria	129
4.3.2	Correctness and sequentialization	138

5	Bracketed Pregroups	153
5.1	Compact bilinear logic and pregroups	155
5.1.1	From the Lambek Calculus to Pregroups	155
5.1.2	Compact Bilinear Logic I	158
5.1.3	Compact Bilinear Logic II	161
5.1.4	The Cut Elimination Theorem for \mathbf{CBL}_{\leq}	165
5.2	β -pregroup grammars	167
5.2.1	Introducing β -brackets	167
5.2.2	Introducing β -antibrackets	174
5.3	Compact β -bilinear logic	177
A	Symmetrical Calculi	185
A.1	Lambek-Grishin Calculus \mathbf{LGC}	185
A.2	Displayed Lambek Calculus $\varphi\text{-}\mathbf{LC}_{\mathbf{d}}$	189
A.3	Displayed Multimodal Lambek Calculus $\sigma\text{-}\mathbf{LC}_{\mathbf{d}}$	191
A.4	Proof net theory for $\sigma\text{-}\mathbf{LC}_{\mathbf{d}}$	197
A.5	Displaying \mathbf{LGC}	205
A.6	Proof net theory for \mathbf{LGC}	211
	Conclusions	213

List of Figures

1.1	An example of an \mathcal{NL} tree	20
1.2	Abstract vs. graph theoretic \mathcal{NL} trees	21
1.3	A tree context and a tree in context	22
1.4	Trees in a pure tree context $\sigma[H_1, \dots, H_m]$	22
1.5	Commutative and associative rules for trees.	23
1.6	An example of an \mathcal{NL} seaweed	25
1.7	Abstract vs. graph theoretical \mathcal{NL} seaweeds	27
1.8	Trees in a pure seaweed context $\alpha[H_1, \dots, H_m]$	27
1.9	Commutative and associative rules for seaweeds	28
1.10	Links in an \mathcal{NL} partial proof structure	31
1.11	Proof structures: same doors, different identity links	32
1.12	Pseudostructures: same doors, different geometry	36
1.13	An application of the contraction rule	39
1.14	An application of structural rules and contraction	39
1.15	Examples of final links and their removal	40
1.16	A bridging link in an \mathcal{NL} structure	41
1.17	A par link and its left and right switchings	43
1.18	Splitting links: a final tensor link and a cut link	46

2.1	Examples of φ - LL seaweeds	55
2.2	φ - LL derivations: identity, cut and logical rules	56
2.3	φ - LL derivations: structural rules	57
2.4	Graphical format of a φ - LL derivation	59
2.5	Examples of φ - LC trees	61
2.6	Identity, cut and logical rules of φ - LC	62
2.7	Structural rules	62
2.8	Links of a φ - LL proof structure	67
2.9	Inductive definition of the φ - LL proof structure (\mathcal{D})	68
2.10	Links for φ - LC	74
2.11	An example of a grafted tree	78
3.1	Rules of Φ - LL	84
3.2	Rules of Φ - LC calculus	89
3.3	Links for Φ - LC	92
3.4	Rules of Φ - LLb	94
3.5	Rules of the Φ - LCb calculus	96
3.6	Ternary links for the Lambek calculus with brackets	101
3.7	Switchings of a ternary par link	101
3.8	An incorrect LCb proof structure	103
4.1	Discontinuous operators: an informal introduction	114
4.2	Rules of BDLC	118
4.3	LC links with parameter edges	119
4.4	Discontinuous links with parameter edges	120
4.5	An incorrect BDLC proof structure	121

4.6	A \wp -link, its left and right switching and a hyperswitching . . .	122
4.7	Reading a theorem off the proofnet	122
4.8	An example of a proof net for a discontinuous idiom	123
4.9	Proof nets for ‘everyone loves someone’	124
4.10	Rules of \mathbf{HC}_e^ω	128
4.11	The hyperforgetful translation f	129
4.12	\mathbf{HC}_e^ω links (first part)	131
4.13	\mathbf{HC}_e^ω links (second part)	132
4.14	Edges ordering in a hyperstructure	133
4.15	Presequent instructions on an output and an input conclusion	137
5.1	Sequent rules for the Lambek calculus \mathbf{LC}	156
5.2	Non-commutative Multiplicative Linear Logic	157
5.3	\mathbf{CBL} : two-sided presentation	159
5.4	\mathbf{CBL} : one-sided presentation	162
5.5	Compact β -bilinear logic	178
5.6	Compact β -bilinear logic	179
A.1	\mathbf{LGC} rules	187
A.2	Deriving (G1) and (G1’)	187
A.3	Minimal set of rules	190
A.4	Structural rules of $\varphi\text{-}\mathbf{LC}_d$	191
A.5	An input and an output directed tree	197
A.6	Configurations and polar trees	198
A.7	Abstract and graph-theoretical polar trees	198
A.8	Polar and graph-theoretical $\sigma\text{-}\mathbf{LC}_d$ seaweeds	199

A.9 Directed logical links.	201
A.10 A directed proof structure and its pseudostructure	202
A.11 Contractions in directed pseudostructures.	203
A.12 Structural rules of \mathbf{LGC}_d	205
A.13 Alternative set of structural rules of \mathbf{LGC}_d	205
A.14 Grishin rewriting rules.	212

Abstract

This thesis studies proof nets for several variants of the Lambek Calculus and their classical extensions. In the first part, *Beyond Monomodality*, the focus is on basic issues of the Lambek Calculus with/without Permutation and Associativity, combinations of these variants and the use of unary modalities. The second part, *Beyond Multimodality*, and the Appendix are dedicated to three recent developments of the Lambek Calculus, namely the *Discontinuous Calculus*, *Pregroup Grammars*, and the *Lambek-Grishin Calculus*.

Keywords: sublinear logic, multimodality, proof nets, Lambek Calculus, discontinuity, pregroups, Grishin rules.

Acknowledgments

During the preparation of this thesis, I have lived through the happiest and, alas, the saddest moments of my life. It is to the people that have helped me overcoming these difficult moments that I wish to dedicate these acknowledgments. Researchers that have helped me academically will bear with me if I simply mention them:

Brendan Gillon, Claudia Casadio, Glyn Morrill, Jim Lambek, Mario Piazza, Michael Moortgat, Michele Abrusci, Oriol Valentín, Richard Moot, Roberto Maioli, Wojciech Buszkowski.¹

I had intended their names to appear here in a playful letter addressed to the child that my wife and I were expecting to have in August 2006. But our Carlo was born extremely premature. In just a week we were confronted with the miracle of his coming into this world and the mystery of his passing away. Despite all, I still wish to address him these lines.

Dear Carletto,

Today daddy has finished writing his thesis. It has been a long project, started long before you had been conceived. Indeed, without this thesis I might have never met mom. And without her love, I would not have had the strength to finish it.

Many people have helped me during these years. Glyn has been a careful thesis director and a true friend. Several researchers have explained me their work and ideas. And many more people have helped mom and me when you were with us and in the weeks and months that followed. They

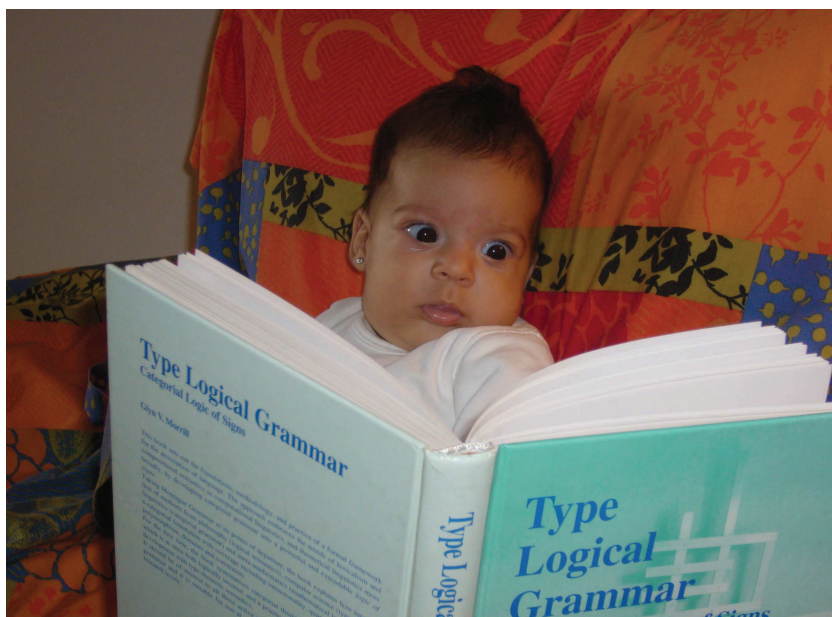
¹I should add to this list, at least, two more professors of my department, namely Ulises Cortés that encouraged and helped me enrolling in the program and Jose Luis Balcázar Navarro, creator –jointly with other colleagues– of the MOISES project (Ministerio de Ciencia y Tecnología TIC2003-04019-C03-01) that supported most of this work. And of course, in this final version, I should add the anonymous reviewers.

have supported us with their care and love. The doctors have made all that was possible for you. The believers have prayed for you. In the most difficult moments, everybody found a good word for us, a hug that gave us strength, a smile that helped us looking forward. Our families, friends and even people that we did not know have let us live through the most intense spiritual moment of our life. We experienced at the same time the fragility of life and the strength of human relationships.

The warmth of those moments gave us strength when we found out that my mom was terminally ill. But grandma was always a courageous and resourceful person. She was sad to depart from us, but grateful for the life she had lived. Her integrity was a last lesson, after a life-long relationship of tenderness, dedication and love.

And now that we have been blessed by the birth of your little sister Dúnia, life has started smiling again at us. But we will always keep you close to our heart, never forgetting your silent, yet powerful reminder: care for your family, for your friends, for the people that you don't even know.

Love, your daddy.



Dúnia, learning the basics of language.

Ai miei cari,
passati e presenti,

nella speranza di poter perpetuare nei figli
l'amore ricevuto dai genitori.

Introduction

Contextualization: Type-Logical Grammars and Lambek Calculus

The central tenet of type-logical grammars ([69, 79, and references therein]) could be stated by the slogan:

“grammaticality = provability”.

Under this view, writing a grammar G based on a logic L consists in modelling a language as a finite set of *lexical items* $LI = \langle \phi, \tau, \sigma \rangle$, where ϕ is the phonological label of LI , τ is a (syntactic) type of the logic L , and σ encodes the semantic information of LI . In this model, a string $\phi_1 \dots \phi_n$ of phonological labels is a grammatical sentence of type t if and only if there are lexical items $\langle \phi_1, \tau_1, \sigma_1 \rangle, \dots, \langle \phi_n, \tau_n, \sigma_n \rangle$ such that $Hyp(\tau_1, \dots, \tau_n) \Rightarrow t$ is a theorem of the logic L , where $Hyp(\tau_1, \dots, \tau_n)$ is a structuring of the types compatible with the linear order of the phonological labels. A strict implementation of the Fregean principle of compositionality of meaning ([39, 95]) can be achieved associating a semantic operation to each (syntactic) inference rule ([120, 121, 92]). In this way, every derivation of a sentence yields deterministically a formula σ encoding its semantic import.

Much of the research in this framework is based on the Lambek Calculus ([57]), an intuitionistic logic sensitive to the number of times hypothesis are used (exactly once) and to their linear order. In its simplest form, $\mathbf{L}(\backslash, /)$, the calculus has only two operators –an implication and a retroimplication–traditionally denoted by a backward and a forward slash. By way of illustration, consider the lexicon of a toy fragment of English, where “Mary” and “Barcelona” have type \mathbf{np} and “visited” has type $(\mathbf{np} \backslash \mathbf{s}) / \mathbf{np}$. The string “visited Barcelona”, analyzed in a traditional structuralist theory as a verbal phrase, is an expression of type $\mathbf{np} \backslash \mathbf{s}$ (just like any intransitive verb). When immediately preceded by “Mary” it yields a grammatical sentence of type \mathbf{s} because $[\mathbf{np}, [(\mathbf{np} \backslash \mathbf{s}) / \mathbf{np}, \mathbf{np}]] \Rightarrow \mathbf{s}$ is a theorem of the calculus. The

associative rule allows to restructure the hypothesis changing the bracketing to $[[\mathbf{np}, (\mathbf{np} \backslash \mathbf{s}) / \mathbf{np}], \mathbf{np}]$. This means that also “Mary visited”, traditionally *not* a constituent, can be given a type, namely \mathbf{s} / \mathbf{np} . Graphically this can be represented in the following way:



The calculus is thus a logic for typing trees identified up to associativity, i.e. a logic for typing lists. It can be defined as intuitionistic logic, from which the following structural rules have been dropped, with the logical (and linguistic) consequences listed below:

- permutation: the linear order of hypothesis (lexical items) is fixed;
- weakening: all the hypothesis have to be used at least once (adding one lexical item to a grammatical sentence will not, in general, yield a grammatical sentence);
- contraction: hypothesis cannot be used multiple times (deleting a duplicate occurrence of a lexical item of a grammatical sentence will, in general, lead to ungrammaticality).

A major consequence of the lack of the latter two structural rules is that conjunction splits into two operators, a multiplicative conjunction introduced (on the right) by a context independent rule, and an additive conjunction introduced by a context sharing rule:

$$\begin{array}{l}
 \text{multiplicative} \quad \frac{\Gamma, A_0, A_1, \Delta \Rightarrow C}{\Gamma, A_0 \times A_1, \Delta \Rightarrow C} \quad \frac{\Gamma \Rightarrow A_0 \quad \Delta \Rightarrow A_1}{\Gamma, \Delta \Rightarrow A_0 \times A_1} \\
 \text{additive} \quad \frac{\Gamma, A_i, \Delta \Rightarrow C}{\Gamma, A_0 \wedge A_1, \Delta \Rightarrow C} \quad \frac{\Gamma \Rightarrow A_0 \quad \Gamma \Rightarrow A_1}{\Gamma \Rightarrow A_0 \wedge A_1}
 \end{array}$$

The Lambek calculus $\mathbf{L}(/, \backslash, \times)$, with the implications and product, had been designed having in mind its interpretation in language models. A semigroup model is built from a semigroup (G, \cdot) , the subsets of which serve as interpretations of the types. The semigroup product is lifted at the level of operation between subsets, and two directional residuation operations can be defined and used to interpret the slash operators of the calculus:

- $A \cdot B = \{a \cdot b : a \in A, b \in B\};$
- $A \setminus B = \{c \in G : \forall a \in A. a \cdot c \in B\};$
- $B / A = \{c \in G : \forall a \in A. c \cdot a \in B\}.$

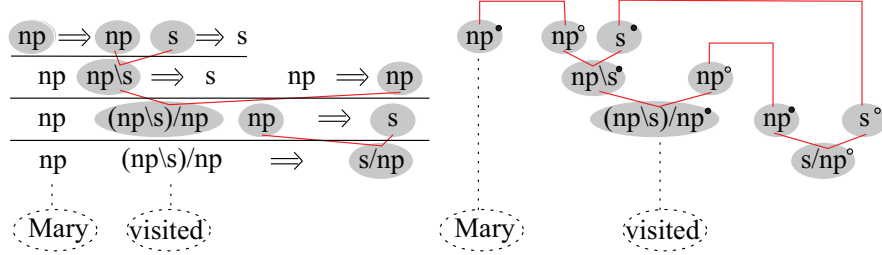
If G is freely generated by a set X (i.e. G is the set of finite strings of elements of X and the product is concatenation), then one speaks of a language model. The Lambek Calculus is complete with respect to these models, both in its version without product ([14]) and with product ([97, 98]). It enjoys, even with the addition of \wedge and its de Morgan dual, the Cut Elimination Property ([57]), which entails decidability. This is because all rules, except the eliminable cut rule, have in their premisses sequent only subformulae of the conclusion sequent and thus the proof search is finite. For $\mathbf{L}(\setminus, /)$ ([114]) and $\mathbf{L}(/, \setminus, \times)$ with or without empty premisses ([99]) the derivability problem is NP-complete. While $\mathbf{L}(\setminus, /)$ and $\mathbf{L}(/, \setminus, \times)$ are weakly equivalent to context free grammars ([96, 100]), unconstrained addition of the additive conjunction and disjunction, increases the recognizing power of the calculus ([50]).

The advent of Linear Logic and Proof Net Theory

Interest in the Lambek Calculus increased in the eighties for two reasons. On one hand, in theoretical linguistics, there has been a progressive convergence toward lexicalism with the emergence of, among other theories, *Lexical Functional Grammar* ([12, 13]) and *Head-Driven Phrase Structure Grammar* ([104, 105, 106]), and with the adoption in the last theoretical proposals by Chomsky of a notion of derivation ([24]). On the other hand, since Girard's seminal work on *Linear Logic* ([42]), the Lambek Calculus can be seen as the intuitionistic (and exponential-free) fragment of *Cyclic Linear Logic* ([109]), a version of *Linear Logic* with a restricted form of commutativity ([123]). As a consequence, properties of the Lambek Calculus can be studied in the environment of linear logic, which, being classical, displays more symmetries and is often easier to investigate. Thus, new concepts of this theory can be imported. For instance, it is possible to import from Girard's theory the notion of proof net ([110]). A proof net, roughly speaking, is a graph that retains at each step of a derivation only the type just introduced and its immediate subtypes, the context of application of a rule being recoverable from the structure of the graph. An advantage is that proof nets capture better the essence of a theorem of the Lambek Calculus because they identify exactly those derivations that are equal up to differences in rule ordering which do not lead to a different semantic interpretation. Moreover, the semantic interpretation can be read off the proof net ([30]). A major consequence of

the introduction of proof net theory is that it offers a way to treat logical problems using graph theoretical tools, hence the word *geometry* in the title of this thesis.

To illustrate the construction of a proof net, consider a derivation of the type $\mathbf{s/np}$ associated to the string “Mary visited” (see below). Of the derivation retain only the highlighted types, marking them with the *polarity* symbol \bullet (\circ) whenever they are an hypothesis (conclusion) of a sequent. Join moreover with an edge the two types that come from identity axioms (the first line in each branch of the derivation); similarly, join the type introduced by each rule with its immediate subtypes. The informed reader will recognize in the input symbol a disguised negation of linear logic. Indeed, proof nets of the Lambek Calculus are essentially proof nets of Cyclic Linear Logic labelled by intuitionistic types.



The graph thus constructed consists of a list of trees that are obtained unfolding the types labelling their roots and whose leaves are connected by edges, the *identity links*, joining atoms of opposite polarity. Among such graphs, called *proof structures*, those that correspond to a derivation are characterized by correctness criteria, such as the Danos-Regnier criterion for Linear Logic ([25]) which is a reformulation of Girard’s long trip condition ([42]). For instance, a proof structure of the Lambek Calculus is correct if it is planar, it has exactly one output conclusion and each of its elementary cycles goes through both immediate subtypes of (at least) a type introduced by a unary inference rule, i.e. a rule with just one sequent premiss ([110]). In the example above, the proof structure meets the criterion because its unique cycle contains the premisses of the (output) type $\mathbf{s/np}$.

Given a sentence (output) type and a list of (input) types corresponding to lexical items and considering their unfoldings into trees, one can attempt to build incrementally left to right a correct proof structure by adding identity links always preserving planarity and avoiding the creation of incorrect cycles. Moreover, a complexity profile can be associated to such correct graphs. It counts, at any point P between two trees, the number of identity links that join leaves of trees preceding P with leaves of trees following

P. Interestingly, for a number of English constructions there appears to be a direct correlation between the complexity profile of the proof net associated to their analyses in the Lambek calculus and their psycholinguistic processing complexity ([48, 81]). This yields an implementation, in a logical framework, of the Dependency Locality Theory ([40]) which, in its simplest version, states that processing complexity is directly proportional to the number of unresolved syntactic dependencies. This is particularly evident in the case of center embedded relatives ([23]):

The cheese that the rat ate stank.

? The cheese that the rat that the cat saw ate stank.

?? The cheese that the rat that the cat that the dog chased saw ate stank.

The dramatically decreasing acceptability of sentences obtained iterating center embedding can thus be accounted for in terms of performance rather than competence. They are considered grammatical, but involving a more and more difficult computation that gets rapidly beyond working memory capacity.² A study shows that local minimalization of the complexity profile appears to be a good heuristic in incremental building of proof nets ([118]). Actually, if the Dependency Locality Theory is correct, one could even put a maximal limit to the complexity without losing any acceptable sentence. Performance of aphasics observed in experimental conditions ([20]) have also been analysed in this framework ([85]), suggesting that their speech problems might depend on deficits of working memory capacity rather than on partial loss of (implicit) knowledge of language.

Issues of generation

Clearly, the Lambek Calculus is not flexible enough to deal with all linguistic phenomena. In particular, it is unable to treat a number of constructions whose analysis would involve, in a derivation, manipulation of non-adjacent types, as in the following cases:

- discontinuous constituents, such as phrasal verbs: John called her up.
- non-peripheral extraction, such as relativization of the object of a transitive verb followed by an adverb: ...that John met _____ yesterday.

²An acceptability judgement is an empirical observation about the (degree of) perceived correctness, by speakers, of a sentence. Grammaticality refers to conformity with a theory. Unfortunately, much of linguistic literature uses the term grammaticality judgement as a synonym for acceptability judgement.

Nor can it deal with phenomena seemingly involving multiple use of the same resource, as in constructions containing *parasitic gaps* ([111, 113]), exemplified here by a sentence where the relative pronoun plays the role of the direct object of both transitive verbs:

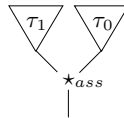
- This is the paper that John filed _____ without reading _____.

On the other hand, certain classes of unacceptable sentences would be considered grammatical, such as the following examples displaying *island constraints* ([111]):

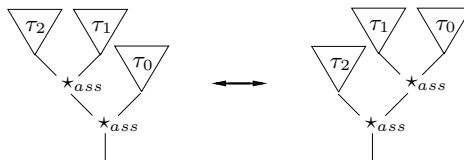
- relativization out of one constituent of a conjunction:
I like the city that Peter wrote about Barcelona and John visited _____.
- questioning a constituent of an adverbial phrase:
Where did I meet John when he was studying at _____?

Beyond Monomodality

To overcome these limitations, several generalizations have been proposed that can all be described as particular instances of a *Multimodal* Lambek Calculus ([94, 45, 69]). A *mode* is a way of combining hypothesis. For example, in the Lambek Calculus, there is just one mode, concatenation of strings. Since concatenation is associative, it follows that the product in the Lambek Calculus is associative as well. More explicitly, one can define an operation \star_{ass} on binary trees whereby two trees τ_0 and τ_1 can be joined in the following way:

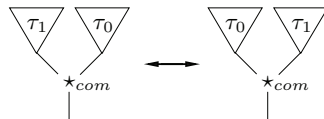


This operation yields an associative mode, provided that the logic is enriched with a rule that allows to restructure trees in the following way:



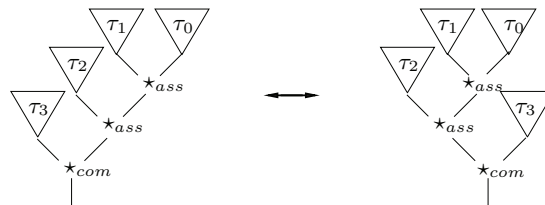
Indeed, it is easy to show that the above structural rule, together with the \times introduction rules mentioned earlier, allows to prove the equivalence of the types $(A \times_{ass} B) \times_{ass} C$ and $A \times_{ass} (B \times_{ass} C)$. In turn, the latter equivalence, together with the cut rule, is sufficient to prove the above structural rule. The advantage in defining the calculus with the structural rule on trees is that in this way the cut rule is still eliminable. For the same reasons the type, say, $(A \times_{ass} B) \times_{ass} C$ can be proved in this mode starting from hypothesis A , B , and C structured either as $[_{ass}[_{ass}A, B], C]$ or $[_{ass}A, [_{ass}B, C]]$. Dropping from the calculus the associative rule, i.e. considering a non-associative mode, only the first structuring would be acceptable. The logic thus obtained, the Non-Associative Lambek Calculus **NLC**, has a particular theoretical importance, because it is the most restrictive variant of the Lambek Calculus.

Other modes can be defined. In the Lambek Calculus with permutation **LCP**, there is one commutative mode \star_{com} which enjoys, beside the associative rule, also the commutative rule expressed below:

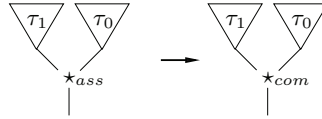


Given this structural rule, not only the product turns out to be commutative, but also the implication and the retroimplication coincide.

Various modes can coexist in a single calculus, each one with an associated product and the implications. The definition of the logical operations is constant across modes. It is the structural rules that vary. They determine the properties of the operations of a particular mode and the possible structuring of the hypothesis of a theorem. To give but a simple example, assume that the type given above for “visited” is in the associative mode and give to “yesterday” the type $s \setminus s$ in the commutative mode. Then both sentences “yesterday Mary visited Barcelona” and “Mary visited Barcelona yesterday” will be accepted as grammatical. This is because the following two structurings of the lexical items are equivalent, thanks to the commutative rule:

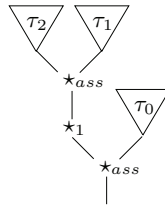


Different modes can also communicate, e.g. with the structural rule of *entropy*, that allows to disregard sensitivity to order ([26, 2, 112]):

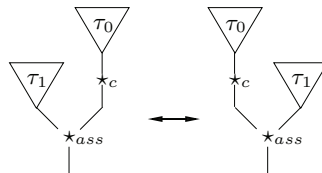


One can also introduce *unary* modes. A unary mode \star_j , simply lifts a tree by adding at its root a unary branching node marked by \star_j . It comes with two operations, denoted by $[j]$ (bracket) and $[j]^{-1}$ (antibracket), that are the unary counterparts of product and implication.³

If a unary mode that licenses no structural rule intervenes between two instances of an associative mode, as in the following picture, then it breaks the configuration and the associative rule can no longer be applied and accounts of island constraints become possible:



On the other hand, a unary mode can be used to license locally a structural rule, as in the following example where the unary mode c allows permutation in an otherwise non-commutative associative environment.



This is useful to account for non-peripheral extraction. The assignment to the relative “that” of the type $\mathbf{rel}/(\mathbf{s}/[\mathbf{c}][\mathbf{c}]^{-1}\mathbf{np})$ allows the \mathbf{np} gap to occur anywhere in the clause following the relative. On one hand, the bracket $[\mathbf{c}]$

³Brackets and antibrackets correspond to the modalities that are often denoted in the literature, respectively, as \diamond and \square^\perp (see, e.g., [54]).

forces the above structuring in which commutativity can be applied. On the other hand, once the restructuring has taken place (or if there is no need for it) the bracket can be removed by the unary version of *modus ponens*, namely the theorem $[c][c]^{-1}x \Rightarrow x$, valid for any type x .

Other unary and binary modes can be defined and, potentially, any rewriting rule for maximally binary branching trees can be considered as a structural rule. Such calculi are so powerful as to be Turing complete in their weak generative capacity ([21]). An interesting restriction on structural rules has been proposed in order to obtain a completeness result ([53]). Only those structural rules are admitted that are weak Sahlqvist, i.e., loosely speaking, that do not contain any form, however disguised, of weakening or contraction. This condition has also the effect of guaranteeing decidability, because the search space is finite. It is not clear, however, whether for linguistic purposes it is too strong a restriction.

Beyond Multimodality

New trends have emerged since the mid 90's to avoid an excessive, ad hoc, use of multimodality.⁴

Joachim Lambek has advocated for a technically simpler approach. Moving from logic to algebra, he has proposed ([60, 61, 62, and references therein]) that the structure of pregroups, resulting from the algebraization of Compact Bilinear Logic ([17]), could be used as a more flexible version of the Lambek Calculus. This new theory seems to go a long way toward a truly minimalist proposal, in that it is formally very simple and does not use theoretical constructs such as structuring of resources, traces, etc.

The basic idea is that syntactic types are the concatenation of simple categories s, n, \dots which are eventually *negated*, possibly more than once, using either a left negation ($s^l, n^l, \dots, s^{ll}, n^{ll}, \dots$) or a right negation ($s^r, n^r, \dots, s^{rr}, n^{rr}, \dots$). A string $\phi_1 \dots \phi_n$ is grammatical if for each ϕ_i there is a type τ_i such that the string $\tau_1 \dots \tau_n$ can be simplified to the symbol s for sentence using instances of the following rules: $a^l a \leq \emptyset$ and $aa^r \leq \emptyset$. The connection between the Lambek Calculus and pregroups is very simple. Essentially one has just to replace the implicative types a/b and $b \backslash a$ respectively by the two formulas ab^l and $b^r a$, thus reducing the two schemata for modus ponens that characterize the Lambek Calculus to the derivations $ab^l b \leq a$ and $bb^r a \leq a$.

⁴The following list of new trends is not exhaustive, since we have restricted our attention to those proposals that came equipped with a theory of proof nets. Their relation with other theories –such as *Abstract Categorical Grammar* ([27, 32]), *Lambda Grammar* ([93]), *Convergent Grammar* ([103, 29])– is left for further research.

Proof structures are reduced to sets of identity links, for which the only correctness criterion is that they do not cross. Although grammars based on pregroups extend those based on the Lambek Calculus, they are still weakly equivalent to context free grammars ([16, 19]). They are appealing for their elegance, but they do not have the flexibility of the Lambek Calculus enriched with modalities and thus there are again problems in dealing with island phenomena, discontinuous constituents, etc.

The latter issue has been central to the reflexion that led Glyn Morrill ([86, 82, and references therein] to the proposal of the Hypersequent Calculus.⁵ The basic idea is that a logical type A of sort n can be realized phonologically by a list of $n + 1$ components, i.e. the type A has a phonological label that displays n points of discontinuity:

$$\boxed{A_0} \quad \boxed{A_1} \quad \cdots \quad \boxed{A_n}$$

In this setting, the product of the Lambek Calculus is still interpreted as concatenation:

$$\boxed{A_0} \quad \boxed{A_1} \quad \cdots \quad \boxed{A_n} \boxed{B_0} \quad \boxed{B_1} \quad \cdots \quad \boxed{B_m}$$

But it becomes also possible –indeed natural– to consider, for any positive integer i , a wrapping product \odot_i , an operation that is interpreted as concatenation of the second factor among the $(i-1)^{th}$ and the i^{th} component of the first factor:

$$\boxed{A_0} \quad \cdots \quad \boxed{A_{i-1} B_0} \quad \boxed{B_1} \quad \cdots \quad \boxed{B_m A_i} \quad \cdots \quad \boxed{A_n}$$

Each wrapping product comes with a pair of residuated operators. Introducing a suitable notation, each family of a product and its residuated pair can be defined by a set of six rules that look formally like the usual rules of the Lambek Calculus $\mathbf{L}(/, \backslash, \times)$. And since each rule is based only on concatenation as a way to structure the types, the calculus is truly monomodal. In this sense, it can be seen as a minimal extension of the Lambek Calculus that allows to deal with discontinuity. Issues of completeness have been studied, for fragments of the calculus that allow for only one point of discontinuity, in

⁵A word of caution is necessary here. The term *hypersequent* had already been used in the mathematical literature ([3, 107]), but in a different sense.

[119]. As for the recognizing power of the calculus, a lower bound has been given for the *Displacement Calculus* \mathbf{D} ([87]), a variant of the Hypersequent Calculus that comprises units for concatenative product and for wrapping product. The calculus \mathbf{D} recognizes the permutation closures of context free languages ([89]).

More recently, Michael Moortgat has considered an extension of the Non-associative Lambek Calculus that could also be regarded as minimal in that it does not rely on multimodality nor does it make use of structural rules (not explicitly, at least). The proposal ([70]) has two basic ingredients. Since the intuitionistic restriction is lifted and sequents are allowed to have multiple conclusions, one can consider for each Lambek operator a dual. Thinking in terms of the language of Non Commutative Linear Logic ([2]), the dual of the product is the par operator, and the dual of an implication, say A/B that can be thought of as $A\wp B^\perp$, is a difference operator $A\oslash B$ that can be thought of as $A\otimes B^\perp$. Moreover, the Grishin rules ([44]) are incorporated in the introduction rules of the implications and their duals. Thus the system is called the *Lambek-Grishin Calculus*, hereafter abbreviated as **LGC**. This system, that has been extended to incorporate unary operators as well as their duals ([22]), is equipped with a continuation semantics ([11]) and with a theory of proof nets ([75]), and it is complete with respect to ternary relational semantics ([55]). A lower bound for the generative capacity of **LGC** has been established in [66], namely the class of languages that are the intersection of a context-free language and the permutation closure of a context-free language.

Structure of the thesis

This thesis studies proof net theory for several of the calculi so far mentioned. In the first part, *Beyond Monomodality*, the focus is on basic issues of the Lambek Calculus with/without Permutation and Associativity, combinations of these variants and the use of brackets. The two chapters of the second part, *Beyond Multimodality*, and the Appendix can be read independently and are dedicated to the three recent trends mentioned earlier. More specifically, the content is divided in chapters as follows:

Chapter 1: I review the relevant literature and explain the notations. In the final section, I propose a characterization of proofnets the substructures of which have at least two conclusions: they correspond to intuitionistic derivations with no empty antecedents.

Chapter 2: it presents a discussion of the diamond of the Lambek Calculi with/without associativity and/or commutativity. It is based on my

publication [36], where I propose that the property of *balance* corresponds, in proofnet theory, to the lack in the logic of the associative structural rule.

Chapter 3: the first part is about multimodal logics obtained combining calculi considered in the previous chapter. I capture the lack of interaction among modalities by the notion of *endomodal* proof structures. As a special case, I characterize proof nets for Moortgaat and Morrill’s biheaded calculus [71]. The second part is about Lambek Calculi with unary modalities. It is based partially on a joint publication with G. Morrill [37], in which Versmissen’s proposal [122] of mimicking unary modalities with two extra symbols is corrected. I furthermore propose an alternative embedding, that works also for commutative logics.

Chapter 4: it improves (in the first part) and generalizes (in the second part) the theory of proof nets for the Discontinuous Lambek Calculus expounded in the joint publication with G. Morrill [83].⁶

Chapter 5: it is about introducing unary modalities in the theory of pre-groups and enriching accordingly the theory of proof nets. It is based on my publication [35], including some new comments to compare it with relevant literature.

Appendix A: the Lambek-Grishin Calculus $\mathbf{LG}_\emptyset + \mathcal{G}^\dagger$ of [70] is reviewed as a special case of a Displayed Lambek Calculus ([43]). A theory of proof nets for such calculi is proposed, where edges are labeled by types and polarities are encoded as directions on the edges.

⁶Without doubts, this chapter would have been very different if we had not been working with O. Valentín on the joint publications [90, 84, 91].

Part I

Beyond monomodality

Chapter 1

Preliminaries

The basic building stones of any theory of natural language are the lexical items, minimal structured clusters of information that establish a connection between the phonological and the semantical level of the analysis. For instance, **HPSG** represents the lexical entry for a proper noun by the following complex matrix ([106, page 358]):¹

$$\left[\begin{array}{l} \text{word} \\ \text{PHON} \end{array} \right. \left. \begin{array}{l} \langle \text{Kim} \rangle \\ \text{SYNSEM} \left[\begin{array}{l} \text{synsem} - \text{struc} \\ \text{SYN} \left[\begin{array}{l} \text{HEAD} \left[\begin{array}{l} \text{noun} \\ \text{AGR} \quad 3\text{sing} \end{array} \right] \end{array} \right] \\ \text{ARG-STR} \langle \quad \rangle \\ \text{MODE} \quad \text{ref} \\ \text{INDEX} \quad \text{i} \\ \text{SEM} \left[\begin{array}{l} \text{RESTR} \left\langle \left[\begin{array}{l} \text{RELN} \quad \text{name} \\ \text{SIT} \quad \text{s} \\ \text{NAME} \quad \text{Kim} \\ \text{NAMED} \quad \text{i} \end{array} \right] \right\rangle \end{array} \right] \end{array} \right] \end{array} \right] \right]$$

However, the role played by the lexicon varies widely across formal grammars. On one end of the spectrum, there are theories –such as the early

¹Needless to say, theories are not uniform in their conception of lexical items, let alone in the formalization of the information they contain. Borrowing from [46], we can say that [...] *there is a stronger and a weaker sense in which the term ‘lexical item’ is employed. In the stronger sense, a lexical item is one whose meaning is not systematically derivable from the meaning of any smaller items it may contain. Thus [...] blackbird is clearly a lexical item in this sense, while black bird equally clearly is not. In a weaker sense, a lexical item is one that is not formed by a general, productive rule, such as cancellable [since there are] limitations on the productivity of -able suffixation.*

theories of the generative tradition— where the burden of grammaticality is carried by non-lexical rules. For instance, a (toy!) generative grammar of Catalan might analyse the following sentence on the basis of the rules listed below:

analysis					
	det	n	pp	tv	np
	Les	filles	de la Júlia	estudien	física nuclear.
	The	daughters of the	study	physics nuclear.	
	Julia's daughters		study	nuclear physics.	
rules	$\begin{array}{l} s \rightarrow np \quad vp \\ vp \rightarrow tv \quad np \\ np \rightarrow det \quad n \quad (pp) \end{array}$				

Variants of this analysis are so ingrained in the grammatical tradition that hierarchical structures are often assumed to be the empirical data. When commenting on the example reported above, the *Grammar of Contemporary Catalan* [115] —a work that intends to be as theory neutral as possible— states that *a sentence is not a ‘plane’ structure, rather it is a cluster of hierarchically organized constituents.*²

On the other end of the spectrum we find the type-logical grammars that follow in J. Lambek’s footsteps and that constitute, in my view, the ultimate embodiment of the structuralist conception of natural language. In these theories, the only rules that can be used in checking the grammaticality of a sentence are the very same rules that establish the properties of the metalanguage in which we express the lexical items. Constituents and their hierarchical organization surface as projection of the information compressed in the lexical items. Thus, they are simply theoretical constructions useful in coming up with generalizations.

Under this view, structuring the (syntactic) information contained in a lexical item means combining bits of information by logical operators. Graph-theoretically this can be represented by a tree,³ the nodes of which are

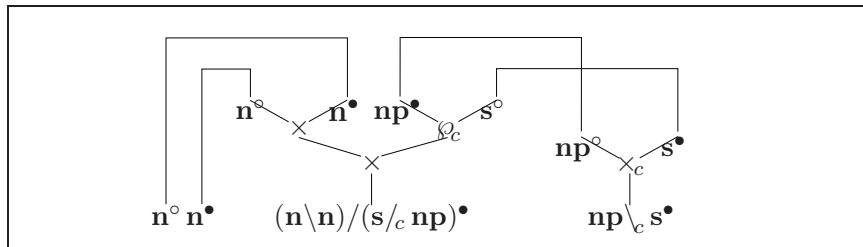
²The translation is mine.

³In much of the mathematical literature ([33]), the term tree is used to refer to an acyclic, connected graph with no distinguished unary node. Following [112, 65], I prefer the term *seaweed* for such a graph, since the metaphor of the tree seems to imply that any tree comes with a root. Therefore instead of rooted trees, I will simply talk of trees.

labeled by the logical operators and the leaves, by the minimal bits of information.

the type of a lexical item	its unfolding as a tree
$(\mathbf{n} \backslash \mathbf{n}) / (\mathbf{s} /_c \mathbf{np})$	

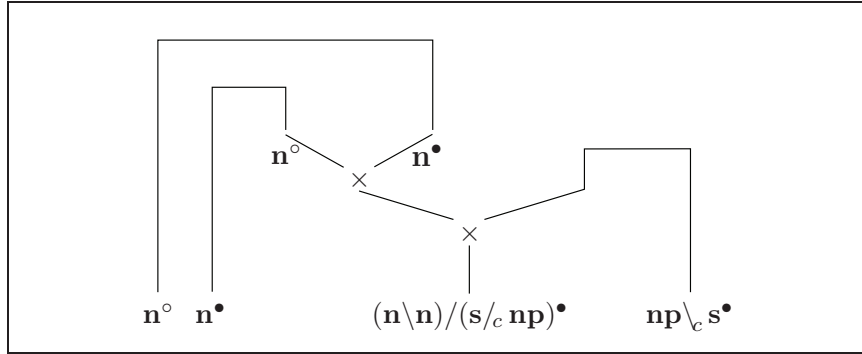
On the one hand, the types decorated by an output polarity (the empty dot) represent information required by the lexical item, i.e. its questions. On the other hand, the types decorated by an input polarity (the full dot) stand for information contained in the lexical item, i.e. its answers. Adding identity links to a multiset of trees yields a proof structure, provided that all answers are justified by a question and that all questions have been answered, as in the following example.



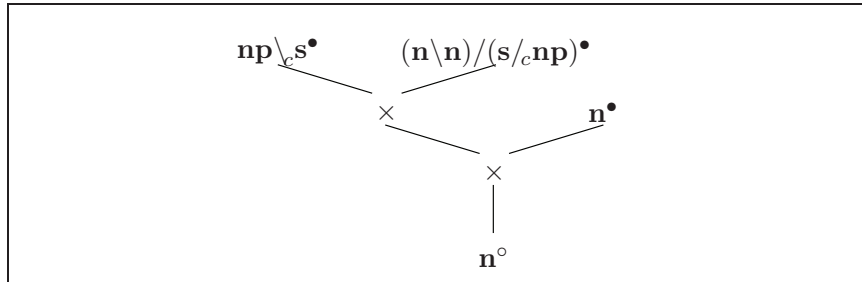
To verify that a proof structure represents a proof of a theorem in a given logic, [74, 108] present a general method based on rewriting rules.⁴ A derivation is successful, and yields the structure of the theorem, if the proof structure can be rewritten as a tree using two classes of rules. On one hand, a proof structure can be modified applying locally the graph-theoretical representation of any structural rule of the logic. In the example above, if the modality marked by the subindex c is commutative, the order of the leaves of the rightmost tree can be altered so as to obtain a planar graph. On the other hand, it is possible to contract a proof structure, removing an elementary planar cycle that goes through exactly one tensor and a par node (of

⁴The method works as long as the structural rewriting rules do not affect the number of leaves in any tree, i.e. as long as the structural rules do not contain any form of weakening or contraction.

the same modality) and connecting directly the two edges that are left dangling.⁵ For instance, continuing the above example, the proof structure can be contracted by removing its unique cycle and connecting the rightmost premise of the central tree and the rightmost input type. This amounts to compacting two pairs answer/question into one more coarse-grained pair, as pictured below:



Once the derivation has removed all cycles, one can look at the graph as a tree rooted in its unique output conclusion. This is the structure of the theorem. The types that constitute its antecedent can be read, right to left, on its leaves.



For many calculi, the problem of sequentializability of proof structures has been studied also from a declarative point of view. Generally, the correctness criteria are expressed on subgraphs of the proof structures and are invariant along successful derivations of the procedural theory reviewed above, hence their necessity. To prove that they are sufficient, one common technique is to adapt the Tensor Splitting Lemma to the logic under consideration and proceed by induction over the complexity of the proof structure. When a proof structure Π can be sequentialized as a derivation of a theorem α ,

⁵There are also contraction rules for unary operators, see Definition 1.18.

the structure of α is implicitly contained in Π . For some logics, it can be derived from a class of subgraphs. Each subgraph of the class might contain the same information, or they might have to be combined by some operation that gathers the relevant information.

The theories so far discussed present an inherent asymmetry, since they deal with proof structures that have only one output conclusion. Removing this restriction one moves from an intuitionistic to a classical environment.

This chapter is divided in two parts. In the first section, we discuss the notion of tree and of seaweed, its classical counterpart. In the second section we review several equivalent definitions of intuitionistic and classical proof structures and a portion of the procedural and declarative theories of correctness that is relevant for our purposes. The chapter ends with the proposal of (DR_2) , a condition that holds necessarily in a proof net that represent classical (intuitionistic) derivations where sequents are formed by at least two types (an output and at least an input type). In particular, this condition, unlike the *no subtending* condition proposed in [31] for the Lambek Calculus, holds also in commutative logics.

1.1 Structuring information

1.1.1 Trees

This section is about a set of labelled finite trees, suitable for a theory of proof nets where logical operators label nodes of the proof structures. As mentioned earlier, I prefer to speak of trees, rather than *rooted* trees, since this metaphor implies –in my opinion– that the root is an intrinsic part of such a graph. The familiar notions of tree context, substitution in a tree context and structural rules are recalled as well. All graph-theoretical definitions are given for finite objects.

Let \mathcal{N} and \mathcal{L} be non-empty disjoint sets. Let $\{\mathcal{N}_1, \mathcal{N}_2\}$ be a partition of \mathcal{N} , i.e. let \mathcal{N}_1 and \mathcal{N}_2 be disjoint sets such that $\mathcal{N} = \mathcal{N}_1 \cup \mathcal{N}_2$. For any $i \in \mathcal{N}_i$, the elements of \mathcal{N}_i will be called modalities of arity i . \mathcal{L} is mnemonic for leaves but also, as will be seen later, for language. \mathcal{N} is mnemonic for nodes.

Graph-theoretical vs. abstract definition of $\mathcal{N}\mathcal{L}$ -trees

Definition 1.1 *An $\mathcal{N}\mathcal{L}$ -tree is an acyclic and connected graph with unary, binary and ternary nodes. Ternary nodes, if there are any, are labeled by*

elements of the set $\{\times_i : i \in \mathcal{N}_2\}$ and come with a cyclic order on the incident edges. Binary nodes, if there are any, are labeled by elements of the set $\{[j] : j \in \mathcal{N}_1\}$. One unary node is called the root and is not labeled; the other unary nodes, of which there is at least one, are called the leaves and are labeled by elements of \mathcal{L} .

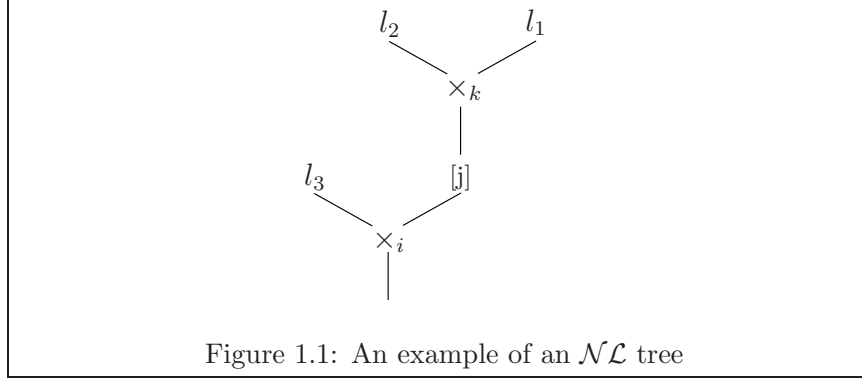


Figure 1.1: An example of an \mathcal{NL} tree

To keep the language simple, we will refer to a leaf labeled by a (definite) occurrence of an element l of \mathcal{L} as a leaf l .

Definition 1.2 *The set of abstract \mathcal{NL} -trees is the smallest set that contains the set \mathcal{L} of leaves and, for all $i \in \mathcal{N}_2$ and $j \in \mathcal{N}_1$, is closed under the binary operator $\tilde{\times}_i$ and the unary operator $\tilde{[j]}$, i.e. abstract \mathcal{NL} trees are defined by induction in the following way:*

$$\tau = \mathcal{L} \mid (\tau \tilde{\times}_i \tau) \mid \tilde{[j]} \tau.$$

The binary and unary operators are also said to label the (non-terminal) nodes of the tree.

For simplicity, we will drop outer parentheses. For example, we will write simply $(A \tilde{\times}_i B) \tilde{\times}_i C$ instead of the more cumbersome $((A \tilde{\times}_i B) \tilde{\times}_i C)$.

The bijective correspondence between abstract and graph-theoretical \mathcal{NL} trees is explained in Figure 1.2. Given the straightforward nature of the correspondence, we will freely switch between the two representations.

\mathcal{NL} -tree contexts and substitutions therein

Let $\mathcal{H} = \{H_n : n \in \mathbb{N}\}$ be a set disjoint from \mathcal{L} . Set $\mathcal{L}' = \mathcal{L} \cup \mathcal{H}$. \mathcal{H} is mnemonic for hole, i.e. a distinguished leaf that appear in a context and

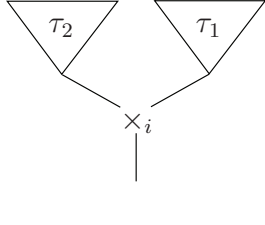
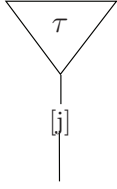
\mathcal{NL} -tree	trivial	non-trivial	
abstract	l	$\tau_1 \tilde{\times}_i \tau_2$	$\tilde{[j]} \tau$
graph theoretic	l 		

Figure 1.2: Abstract vs. graph theoretic \mathcal{NL} trees

that can be substituted by any tree.

Definition 1.3 An \mathcal{NL} -tree context with m holes ($m \geq 0$) is an \mathcal{NL}' -tree $\sigma[H_1, \dots, H_m]$ where m leaves are labeled by H_1, \dots, H_m and the other leaves, if there are any, are labeled by elements of \mathcal{L} . A leaf labeled by an element of \mathcal{H} is called a hole, a leaf labeled by an element of \mathcal{L} is said to be proper. An \mathcal{NL} -tree context is pure if it has no proper leaves.

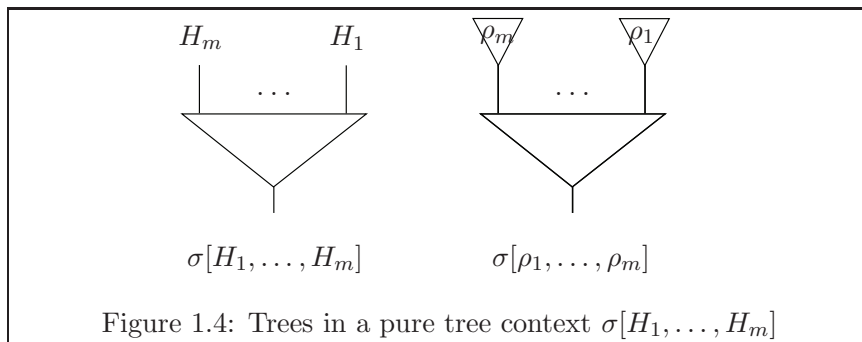
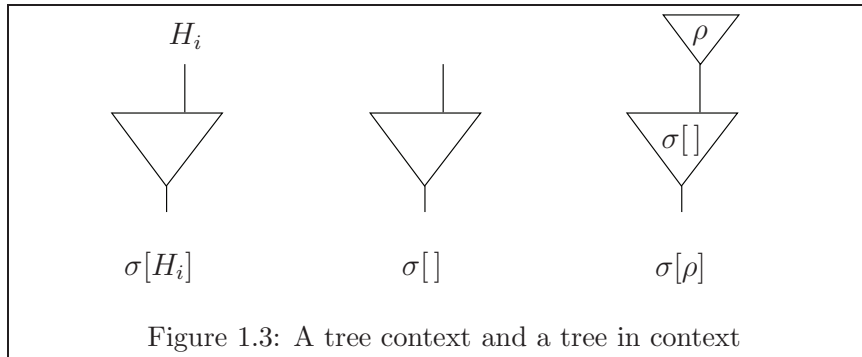
The notation $\sigma[H_i]$ will be used to denote an m -holed context, when we want to focus on the hole labeled by H_i . Removing the label H_i from $\sigma[H_i]$ yields a graph that will be denoted by $\sigma[\]$.

In the following, contexts are always assumed to have at most one hole, unless otherwise specified. Clearly, a context with no holes is a tree.

The result of substituting a context ρ in the H hole of a context $\sigma[H]$ is the context $\sigma[\rho]$ defined as follows:

- if $\sigma[H] = H$, then $\sigma[\rho] = \rho$;
- if $\sigma[H] = \xi[H] \tilde{\times}_i \tau$, then $\sigma[\rho] = \xi[\rho] \tilde{\times}_i \tau$;
- if $\sigma[H] = \tau \tilde{\times}_i \xi[H]$, then $\sigma[\rho] = \tau \tilde{\times}_i \xi[\rho]$;
- if $\sigma[H] = \tilde{[j]} \xi[H]$, then $\sigma[\rho] = \tilde{[j]} \xi[\rho]$.

The tree $\sigma[\rho_1, \dots, \rho_m]$ is the result of substituting, for any $i \in \{1, \dots, m\}$, a tree ρ_i in the hole H_i of $\sigma[H_1, \dots, H_m]$.



Structural rules for \mathcal{NL} -trees

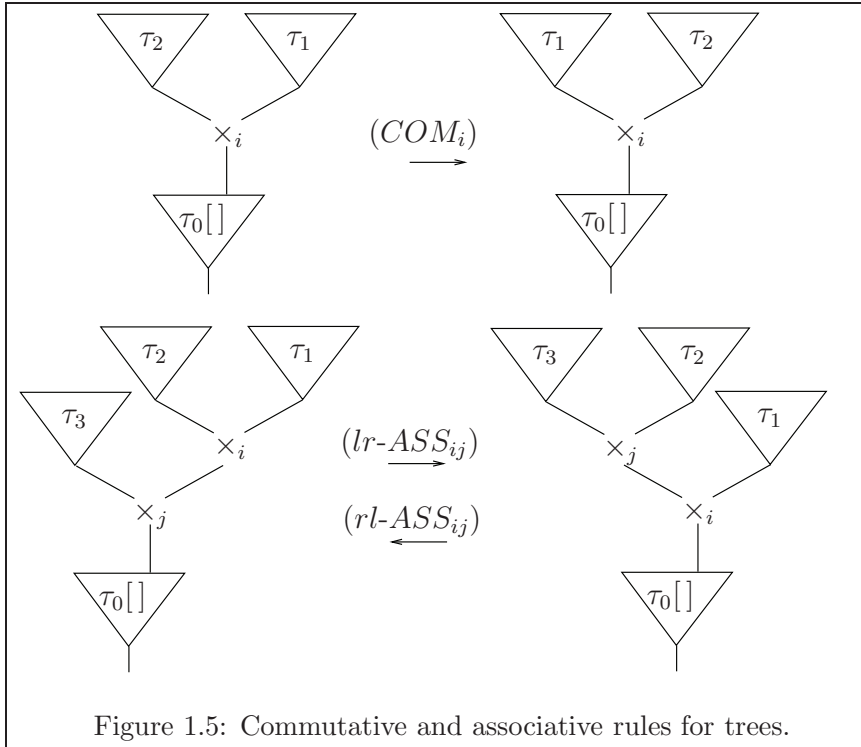
A structural rule is an ordered pair

$$s = \langle \sigma_1[H_1, \dots, H_m], \sigma_2[H_1, \dots, H_m] \rangle$$

of pure contexts with m holes labeled by H_1, \dots, H_m , for some $m \in \mathbb{N}$. Any such rule induces a rewriting rule s on \mathcal{NL} -tree contexts, namely:

$$\tau[\sigma_1[\rho_1, \dots, \rho_m]] \xrightarrow{s} \tau[\sigma_2[\rho_1, \dots, \rho_m]]$$

where ρ_1, \dots, ρ_m are arbitrary \mathcal{NL} -tree contexts and $\tau[H_0]$ is the context in which the restructuring takes place and has therefore at least one hole.



Given a set \mathcal{S} of structural rules and two \mathcal{NL} -tree contexts α and β , say that $\beta <_{\mathcal{S}} \alpha$ if there is a rule s in \mathcal{S} such that $\alpha \xrightarrow{s} \beta$. Let $\leq_{\mathcal{S}}$ be the transitive reflexive closure of $<_{\mathcal{S}}$ and let $\approx_{\mathcal{S}}$ be the equivalence relation generated by $\leq_{\mathcal{S}}$.

Definition 1.4 An \mathcal{S} -sugared \mathcal{NL} -tree (context) is an \mathcal{NL} -tree (context) identified up to the equivalence relation $\approx_{\mathcal{S}}$.

By a slight abuse of notation, a sugared tree (context) – i.e. an equivalence class $[\tau]_{\approx_{\mathcal{G}}}$ – will be denoted, whenever this does not lead to confusion, by any of its representant τ .

1.1.2 Seaweeds

I look down there [...] in the darkness,
there's this green trail, [...] algae [...]
And it was, it was just leading me home.

Ron Howard, Apollo 13.

This section is about the notion of *seaweed*, a straightforward generalization of a notion introduced in [28] to define sequents for a classical extension of the Non-Associative Lambek Calculus. The term comes from [112], where it denoted the graphical representation of sequents of Non-Commutative Linear Logic, and it is meant to evocate the image of a *rootless* tree. The notions of seaweed context, substitution in a seaweed context and structural rules are given as well.

Let \mathcal{N} and \mathcal{L} be non-empty disjoint sets. Let $\{\mathcal{N}_1, \mathcal{N}_2\}$ be a partition of \mathcal{N} . Let $\mathcal{H} = \{H_n : n \in \mathbb{N}\}$ be a set disjoint from \mathcal{L} . Set $\mathcal{L}' = \mathcal{L} \cup \mathcal{H}$.

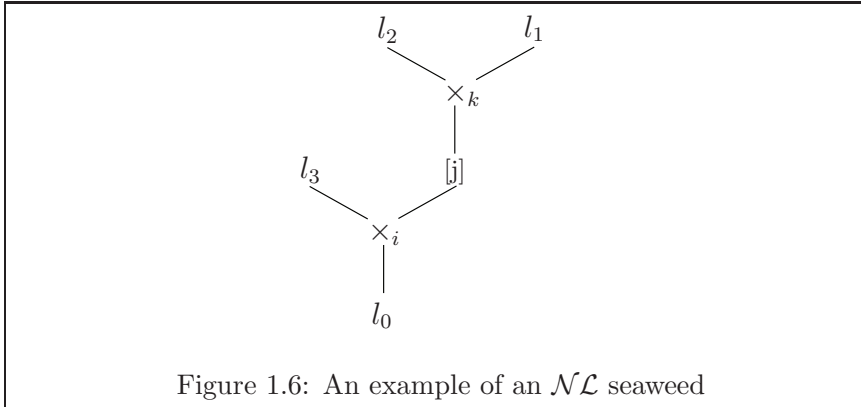
Graph-theoretical vs abstract definition of \mathcal{NL} -seaweed

Definition 1.5 *An \mathcal{NL} seaweed context is an acyclic and connected graph with unary, binary, and ternary nodes. The ternary nodes are labeled by elements of the set $\{\times_i : i \in \mathcal{N}_2\}$. The binary nodes are labeled by elements of the set $\{[j] : j \in \mathcal{N}_1\}$. The unary nodes called (proper) leaves are labeled by elements of \mathcal{L} ; the unary nodes called holes are labeled by distinct elements of $\mathcal{H} = \{H_i : i \in \mathbb{N}\}$. A seaweed has no other nodes and at least one edge. Ternary nodes come with a cyclic order on the incident edges.*

Consider the set of non-ordered pairs (σ_0, σ_1) of abstract \mathcal{NL} -tree contexts with or without holes. On this set let \approx be the smallest equivalence relation such that for all tree contexts τ_0, τ_1 , and τ_2 and for all $n \in \mathcal{N}$:

1. for any binary modality i :

$$(\tau_0 \tilde{\times}_i \tau_1, \tau_2) \approx (\tau_0, \tau_1 \tilde{\times}_i \tau_2) \text{ and}$$



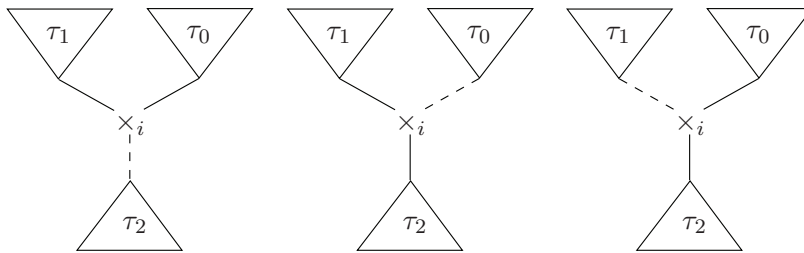
$$(\tau_0 \tilde{\times}_i \tau_1, \tau_2) \approx (\tau_1, \tau_2 \tilde{\times}_i \tau_0);$$

2. for any unary modality j :

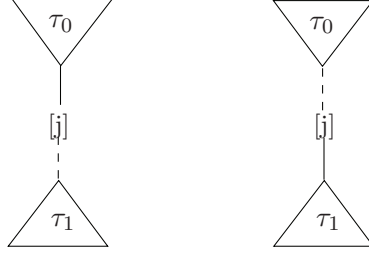
$$([\tilde{j}] \tau_0, \tau_1) \approx (\tau_0, [\tilde{j}] \tau_1).$$

Let us represent graphically a pair (σ_0, σ_1) as the graph obtained joining, by a dashed line, the roots of σ_0 and σ_1 . Then the graphs below represent the pairs that appear in the clauses of the previous definition of the \approx equivalence, namely:

1. for any binary modality i , the graphs $(\tau_0 \tilde{\times}_i \tau_1, \tau_2)$, $(\tau_0, \tau_1 \tilde{\times}_i \tau_2)$, and $(\tau_1, \tau_2 \tilde{\times}_i \tau_0)$ are respectively:



2. for any unary modality j , the graphs $([\tilde{j}] \tau_0, \tau_1)$ and $(\tau_0, [\tilde{j}] \tau_1)$ are respectively:



Denote by $\sigma_0 \star \sigma_1$ the \approx -equivalence class individuated by the pair (σ_0, σ_1) .

Definition 1.6 *An abstract \mathcal{NL} seaweed context $\sigma_0 \star \sigma_1$ with m holes is the \star -product individuated by \mathcal{NL} -tree contexts σ_i with m_i holes ($i = 0, 1$), where $m_0 + m_1 = m$.*

Clearly, there is a bijective correspondence between abstract seaweed contexts and their representations by (graph theoretical) seaweed contexts. Before establishing the correspondence, see Figure 1.7, we introduce some more notations.

In the following, for any binary modality i , the seaweed context $(\tau_0 \tilde{\times}_i \tau_1) \star \tau_2$ will be denoted also by the ternary brackets $[_i \tau_0, \tau_1, \tau_2]$. Observe that for all cyclic permutations $\pi \in (0, 1, 2)$:

$$[_i \tau_0, \tau_1, \tau_2] = [_i \tau_{\pi(0)}, \tau_{\pi(1)}, \tau_{\pi(2)}].$$

Similarly, for any unary modality j , the seaweed context $[_j] \tau_0 \star \tau_1$ will be denoted also by the binary brackets $[_j \tau_0, \tau_1]$. Observe that for all permutations $\pi \in (0, 1)$:

$$[_j \tau_0, \tau_1] = [_j \tau_{\pi(0)}, \tau_{\pi(1)}].$$

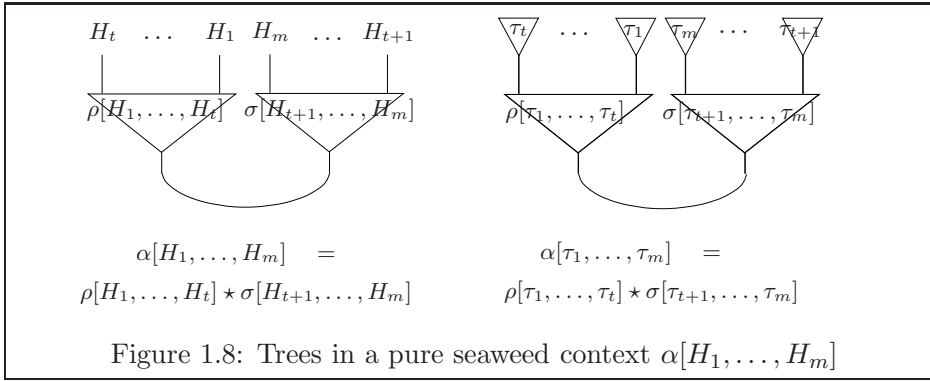
Note that any seaweed context α can be represented by a \star -product of a trivial tree l and a tree τ . Choosing such a representation will be referred to as focusing on the leaf l .

Definition 1.7 *An \mathcal{NL} seaweed context with no holes is called an \mathcal{NL} seaweed. An \mathcal{NL} seaweed context with no proper leaves is called a pure context.*

The result of substituting an \mathcal{NL} -tree context τ into the H hole of a context $\alpha[H]$ is $\alpha[\tau] = \tau \star \beta$, where $H \star \beta$ is the representation of $\alpha[H]$ obtained focussing on H . The result of substituting, for any $i \in \{1, \dots, m\}$, a tree context τ_i for a hole H_i in the context $\alpha[H_1, \dots, H_m] = \rho[H_1, \dots, H_i] \star \sigma[H_{i+1}, \dots, H_m]$ is the seaweed $\alpha[\tau_1, \dots, \tau_m] = \rho[\tau_1, \dots, \tau_i] \star \sigma[\tau_{i+1}, \dots, \tau_m]$.

\mathcal{NL} seaweed	trivial	non-trivial	
abstract	$l_o \star l_1$	$[i \ \tau_0, \tau_1, \tau_2]$	$[j \ \tau_0, \tau_1]$
graph theoretic	$ \begin{array}{c} l_1 \\ \\ l_0 \end{array} $		

Figure 1.7: Abstract vs. graph theoretical \mathcal{NL} seaweeds



Structural rules for \mathcal{NL} -seaweeds

A structural rule is an ordered pair

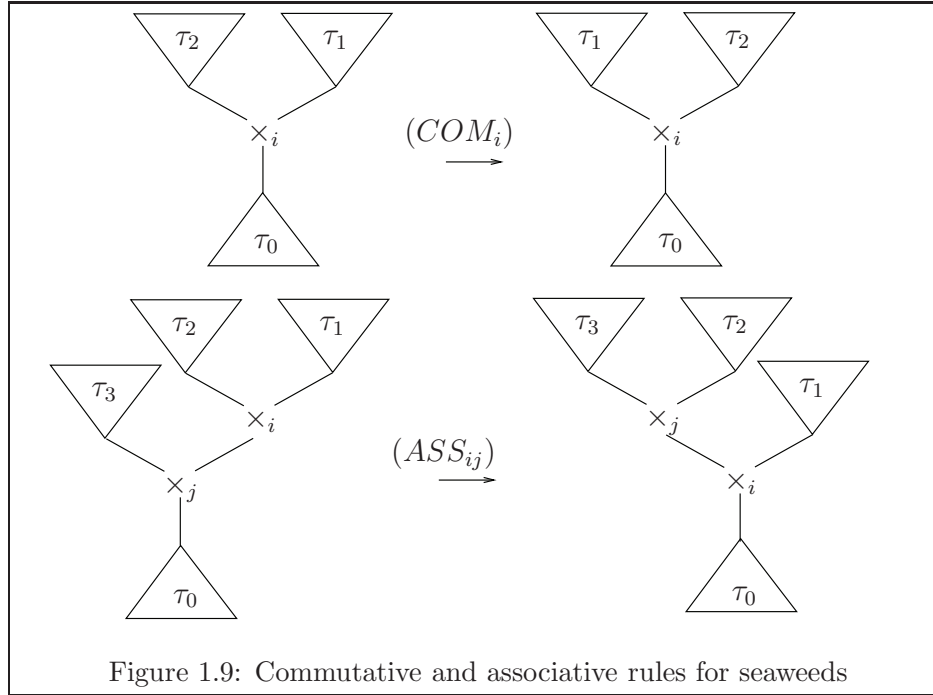
$$s = \langle \sigma_1[H_1, \dots, H_m], \sigma_2[H_1, \dots, H_m] \rangle$$

of pure seaweed contexts with m holes labeled by H_1, \dots, H_m , for some $m \in \mathbb{N}$. Any such rule induces a rewriting rule s on \mathcal{NL} seaweed contexts, namely:

$$\sigma_1[\rho_1, \dots, \rho_m] \xrightarrow{s} \sigma_2[\rho_1, \dots, \rho_m]$$

where ρ_1, \dots, ρ_m are \mathcal{NL} -tree contexts.

Given a set \mathcal{S} of structural rules and two \mathcal{NL} seaweed contexts α and β , say that $\beta <_{\mathcal{S}} \alpha$ if there is a rule s in \mathcal{S} such that $\alpha \xrightarrow{s} \beta$. Let $\leq_{\mathcal{S}}$ be



the transitive reflexive closure of $\leq_{\mathcal{S}}$ and let $\approx_{\mathcal{S}}$ be the equivalence relation generated by $\leq_{\mathcal{S}}$.

Definition 1.8 *An \mathcal{S} -sugared \mathcal{NL} seaweed (context) is an \mathcal{NL} seaweed (context) identified up to the equivalence relation $\approx_{\mathcal{S}}$.*

By a slight abuse of notation, a sugared seaweed (context), – i.e. an equivalence class $[\alpha]_{\approx_{\mathcal{S}}}$ – will be denoted, whenever this does not lead to confusion, by any of its representant α .

Any tree structural rule $\langle \tau_1[H_1, \dots, H_m], \tau_2[H_1, \dots, H_m] \rangle$ yields naturally the seaweed structural rule obtained turning the roots of the trees into an extra hole H_0 , i.e.

$$\langle H_0 \star \tau_1[H_1, \dots, H_m], H_0 \star \tau_2[H_1, \dots, H_m] \rangle .$$

Conversely, any structural rule $s = \langle \sigma_1[H_0, \dots, H_m], \sigma_2[H_0, \dots, H_m] \rangle$ for seaweeds yields a structural rule for trees, simply by choosing to focus on one hole. For instance, choosing to focus on H_0 , the rewriting rule s yields

the following rewriting rule of tree contexts:

$$\langle \tau_1[H_1, \dots, H_m], \tau_2[H_1, \dots, H_m] \rangle .$$

where, for any $i \in \{1, 2\}$, $\sigma_i[H_0, \dots, H_m]$ is thought of as $H_0 \star \tau_i[H_1, \dots, H_m]$ for a suitable tree context $\tau_i[H_1, \dots, H_m]$.

However, there is no bijection between the sets of structural rules, since focusing on different leaves of a seaweed can yield different structural rules for trees. For instance, both left to right associativity (*lr-ASS_{ij}*) and right to left associativity (*rl-ASS_{ij}*) of trees (see Figure 1.5) correspond to one and the same associativity rule (*ASS_{ij}*) for seaweeds (see Figure 1.9). For this reason, the classical extension of a Lambek Calculus enriched with structural rules is not, in general, conservative.

In the next sections we review the procedural theory of correctness, dealing only with structural rules that are not sensitive to polarity. Because of the comments reported in the previous paragraph, we can only consider those intuitionistic calculi the classical extension of which is conservative. For the other calculi we present, following [43], an embedding in a *displayed* calculus. Sensitivity to polarity is regained orienting the edges of the proof structures. This proposal, elaborated independently, has been now superseded by R. Moot's work ([75]) and therefore it is not presented in the main body of the thesis, but in an appendix.

1.2 Checking information

1.2.1 Defining proof structures

This section reviews the familiar notion of proof structure⁶ giving a declarative graph-theoretical definition, a constructive definition based on forests of trees and identity links, and an inductive definition. It presents the notion of pseudostructure, akin to the notion of hypothesis structure ([77]), which is essentially just a graph that retains of a proof structure only its *geometry* and the labeling of the doors. Finally, the notion of underlying linear logic structure is defined and the notion of substructure is reviewed.

For $m \in \{1, 2\}$, let \mathcal{N}_m be a set of modalities of arity m . Consider the following sets:

⁶Two generalizations of the notion of proof structure will be given, respectively, in Chapter 4 and Appendix A: proof structures of the hypercalculus, in which there is a unique modality that underlies different families of operators, and directed proof structures, in which the fundamental difference is that the edges are directed. A unified definition would lead to a more involved wording and no practical advantage.

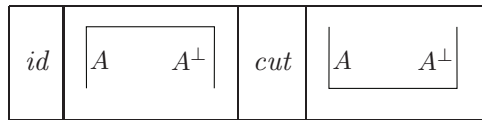
- $\mathcal{N}_\times = \{\times_i : i \in \mathcal{N}_2\}$ of *tensor operators*;
- $\mathcal{N}_\wp = \{\wp_i : i \in \mathcal{N}_2\}$ of *par operators*;
- $\mathcal{N}_{[]} = \{[j] : j \in \mathcal{N}_1\}$ of *bracket operators*;
- $\mathcal{N}_{[]}^{-1} = \{[j]^{-1} : j \in \mathcal{N}_1\}$ of *antibracket operators*.

Graph-theoretical definition of \mathcal{NL} proof structures

Definition 1.9 *An \mathcal{NL} partial proof structure is a connected graph such that:*

- *the (logical) ternary nodes are labeled by elements of $\mathcal{N}_\times \cup \mathcal{N}_\wp$; incident edges are ordered cyclically; one of them is specified to be the conclusion, the left (right) premise is the edge that immediately follows (precedes) the conclusion in clockwise fashion;*
- *the logical binary nodes are labeled by elements of $\mathcal{N}_{[]} \cup \mathcal{N}_{[]}^{-1}$; of the two incident edges, one is the conclusion, the other the premise;*
- *the non-logical binary nodes are labeled by *id* or *cut*; *id* nodes have two conclusions, *cut* nodes have two premises;*
- *the unary nodes are not labeled;*
- *there are no other nodes;*
- *any edge is a premise of at most one node and the conclusion of at most one node; for any node label l , a node labeled by l together with its premises and conclusions is called an l -link;*
- *the edges are labeled by elements of \mathcal{L} ; Figure 1.10 explains the restrictions on the labeling of edges of a link.*

For typographical reasons *id* and *cut* nodes will be represented hereafter by a horizontal line, that can be stretched *ad libitum*:



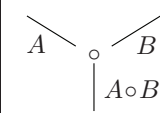
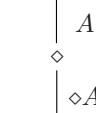
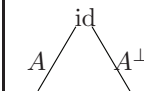
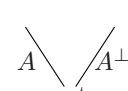
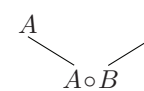



	$\circ \in \mathcal{N}_\times \cup \mathcal{N}_\emptyset$	$\diamond \in \mathcal{N}_[] \cup \mathcal{N}_{[]^{-1}}$	id	cut
link				
premises	A and B	A	none	A and A [⊥]
conclusions	A o B	◇A	A and A [⊥]	none

Figure 1.10: Links in an \mathcal{NL} partial proof structure

There is an alternative syntax for proof structures, in which the elements of \mathcal{L} label the nodes whereas the edges are left unlabeled.⁷ Although the latter notation is more compact, we prefer to use the former because it highlights the structure of the graph. The repertoire of compact links is the following:

	$\circ \in \mathcal{N}_\times \cup \mathcal{N}_\emptyset$	$\diamond \in \mathcal{N}_[] \cup \mathcal{N}_{[]^{-1}}$	id	cut
link				

Definition 1.10 *An open leaf in a partial proof structure is an edge that is the conclusion of no link. A door of a partial proof structure is an edge that is a premise of no link.*

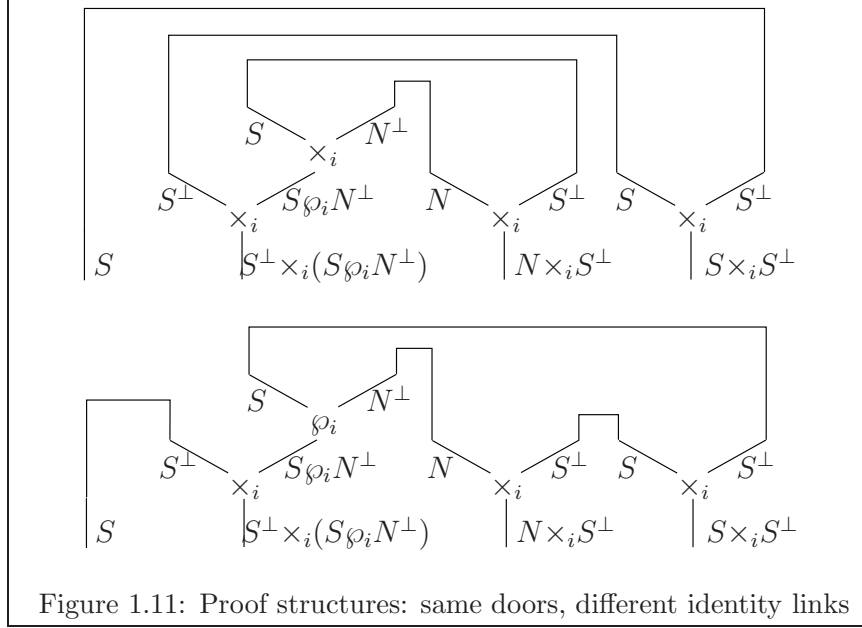
Definition 1.11 *An \mathcal{NL} proof structure is an \mathcal{NL} partial proof structure with no open leaves.*

To keep the language simple, we will refer to an edge labeled by an instance of a type A as the edge A .

Formula trees and \mathcal{NL} proof structures

Proof structures can be built out of (some!) lists of logical types of a language. Each type is unravelled into a formula tree, according to the following

⁷Needless to say, there are also proposals for notions of proof structure that differ in more substantial ways, see e.g. Pentus's notion of proof nets in [99].



definition, and identity and cut links are added so as to obtain a partition of the set of leaves into pairs of atoms and their negations. Different identity linkings can be considered for the same list of formula trees, see Figure 1.11.

Let \mathcal{L} be a subset of a language \mathcal{L}' defined on the basis of a set \mathcal{A} of atomic symbols, the set $\mathcal{A}^\perp = \{A^\perp : a \in \mathcal{A}\}$ of their *negations*, the set $\mathcal{N}_\times \cup \mathcal{N}_\varphi$ of binary operators, and the set $\mathcal{N}_[] \cup \mathcal{N}_[]^{-1}$ of unary operators. Extend the negation metalinguistically to any element of the language \mathcal{L}' setting:

- $F^{\perp\perp} = F$;
- $(F \times_i G)^\perp = G^\perp \varphi_i F^\perp$;
- $(F \varphi_i G)^\perp = G^\perp \times_i F^\perp$;
- $([j]F)^\perp = [j]^{-1}F^\perp$; and
- $([j]^{-1}F)^\perp = [j]F^\perp$.

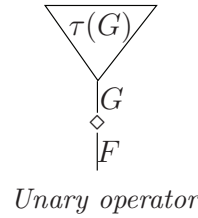
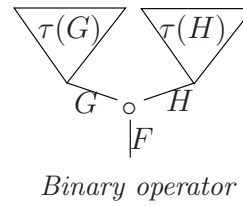
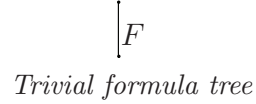
where F and G are any element of \mathcal{L} , $i \in \mathcal{N}_2$, and $j \in \mathcal{N}_1$.

In particular, if \mathcal{N}_1 is the empty set and \mathcal{N}_2 is a singleton set, we say that \mathcal{L} is the language of linear logic. In this case, we drop the modality index

from operators, names of rules, etc.⁸

Definition 1.12 A formula tree $\tau(F)$ is a graph defined as follows by induction on the complexity of a type F :

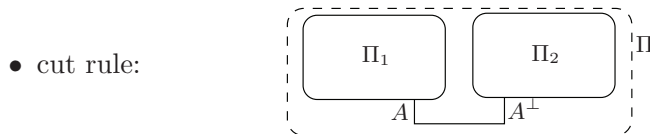
- if $F \in A$, then $\tau(F)$ is the graph that consists of one edge labeled by F and two unary unlabeled nodes;
- if $F = G \circ H$, then $\tau(F)$ is the graph obtained from $\tau(G)$ and $\tau(H)$ joining, around a ternary node labeled by \circ , the edges labeled by G and H with a further edge labeled by F ; moreover, the edges labeled by F , G , and H must be clockwise in this cyclic order around the mentioned node.
- if $F = \diamond G$, then $\tau(F)$ is the graph obtained joining the root of $\tau(G)$ to a binary node labeled by \diamond yielding a tree the root of which is labeled by F .



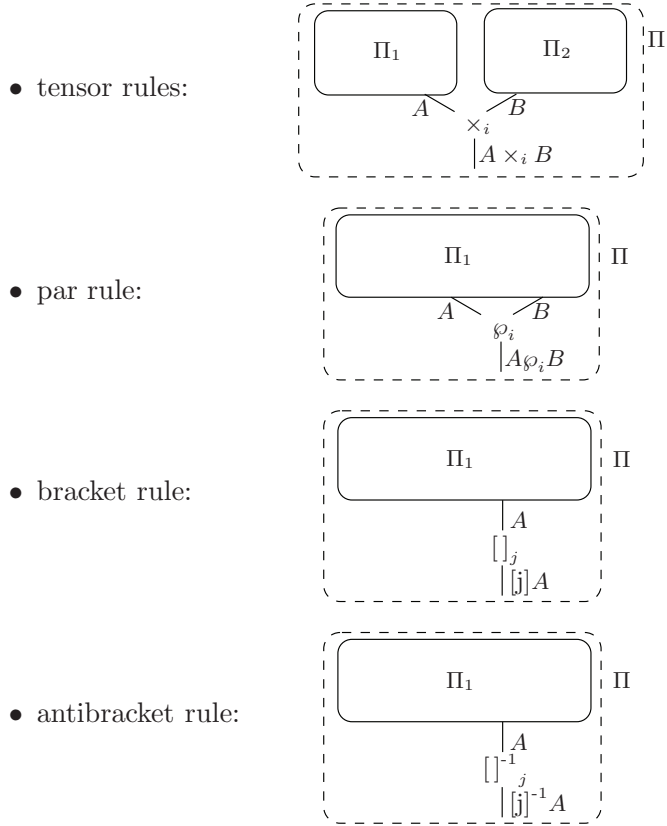
Observe that a tree formula $\tau(F)$ is an \mathcal{NL} -tree exactly when F contains neither par nor antibracket operators.

Inductive definition of \mathcal{NL} proof structures

The following rules, applied to any \mathcal{NL} proof structure Π_h ($h \in \{1, 2\}$), yield an \mathcal{NL} proof structure Π , where $i \in \mathcal{N}_2$ and $j \in \mathcal{N}_1$:



⁸Throughout the thesis, we will deal only with the exponential-free, multiplicative fragment of linear logic.



Definition 1.13 *The set of inductive \mathcal{NL} proof structures is the smallest set that contains \mathcal{NL} identity links and that is closed under:*

- the cut rule;
- for any $i \in \mathcal{N}_2$, the \times_i - and \emptyset_i -rules;
- for any $j \in \mathcal{N}_1$, the $[j]$ - and $[j]^{-1}$ -rules.

Observe that, if \mathcal{N}_1 is the empty set and \mathcal{N}_2 the singleton set, then the set of inductive proof structures coincides with the set of proof nets of multiplicative linear logic (without modalities).

\mathcal{NL} pseudostructures

Pseudostructures are the *trait d'union* between proof structures and seaweeds. In particular, seaweeds are acyclic pseudostructures. When moving

from a proof structure to its *underlying* pseudostructure, one forgets –among other things– which edge of a tensor node is the conclusion. This allows us to modify pseudostructures locally using the same rewriting rules already considered for seaweeds.⁹

Definition 1.14 *An \mathcal{NL} pseudostructure is a connected graph such that:*

- *its ternary nodes are labeled by elements of $\mathcal{N}_\times \cup \mathcal{N}_\emptyset$; incident edges are ordered cyclically; in the case of the nodes labeled by elements of \mathcal{N}_\emptyset , one of the edges is specified to be the conclusion (this is denoted by a tail on the edge);*
- *its binary nodes are labeled by elements of $\mathcal{N}_{[]} \cup \mathcal{N}_{[]^{-1}}$; in the case of the nodes labelled by elements of $\mathcal{N}_{[]^{-1}}$, one of the edges is specified to be the conclusion (this is denoted by a tail on the edge);*
- *unary nodes are labeled by elements of \mathcal{L} ;*
- *there are no other nodes;*
- *edges are not labeled.*

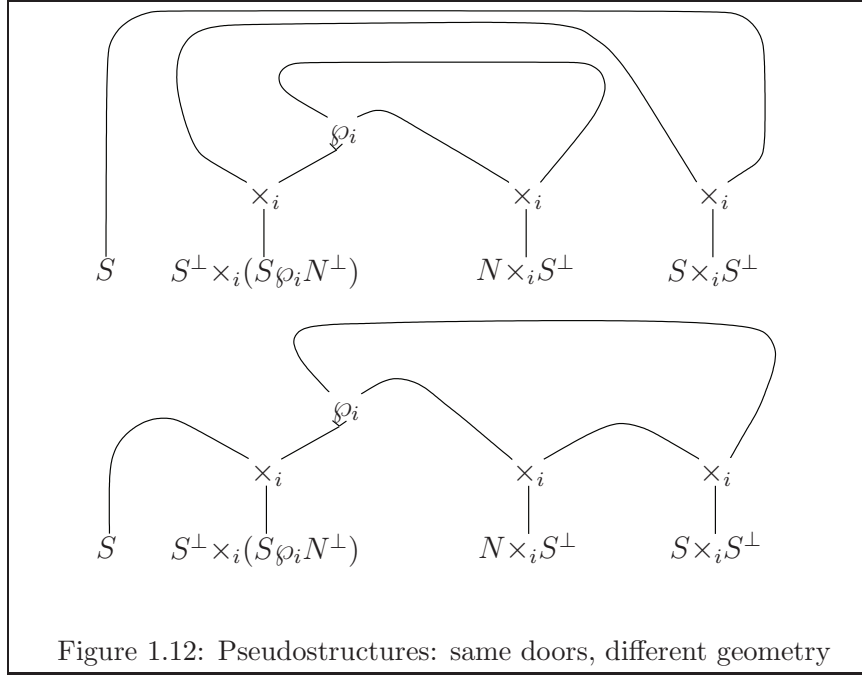
In a pseudostructure, we can still speak of \circ -links as the nodes labeled by a logical operator \circ together with its incident edges.

Given any partial proof structure Π , the *underlying* pseudostructure Π^- is obtained in the following way:

- for any edge incident to a unary node, copy the label of the edge onto the node itself;
- mark the conclusion of any node labeled by an element of \mathcal{N}_\emptyset or $\mathcal{N}_{[]^{-1}}$ with a tail;
- remove edge labels;
- remove any non-logical binary node by connecting directly between them the two incident edges.

Definition 1.15 *An \mathcal{NL} structure is either an \mathcal{NL} proof structure or an \mathcal{NL} pseudostructure.*

⁹The prefix *pseudo* intends to suggest the idea that there are pseudostructures that do not underlie any proof structure.



The underlying linear logic structure

Consider the forgetful function f that associates, to any element of the language \mathcal{L} , an element of the language of linear logic (see page 32) according to the following clauses (where $A, B \in \mathcal{L}$):

- for any atomic A , $fA = A$ and $f(A^\perp) = A^\perp$;
- for any $i \in \mathcal{N}_2$ and $\circ \in \{\times, \wp\}$, $f(A \circ_i B) = fA \circ fB$;
- for any $j \in \mathcal{N}_1$ and $\diamond \in \{[], []^\perp\}$, $f(\diamond_j A) = fA$.

Definition 1.16 *Let Π be an \mathcal{NL} (partial) proof structure. The underlying linear logic (partial) proof structure $f\Pi$ is the (partial) proof structure obtained from Π replacing its links according to the two following instructions (where A and B are elements of \mathcal{L}):*

$$\begin{array}{l}
\textit{id-link} \\
\textit{cut-link} \\
\textit{binary logical links} \\
i \in \mathcal{N}_2, \circ \in \{\times, \emptyset\} \\
\textit{unary logical links} \\
j \in \mathcal{N}_1, \diamond \in \{[], []^1\}
\end{array}
\begin{array}{c}
\boxed{A \quad A^\perp} \\
\boxed{A \quad A^\perp} \\
\begin{array}{c} A \quad B \\ \diagdown \quad \diagup \\ \circ_i \\ | \\ A \circ_i B \end{array} \\
\begin{array}{c} | \\ \diamond_j \\ | \\ \diamond_j A \end{array}
\end{array}
\begin{array}{c}
\rightarrow \\
\rightarrow \\
\rightarrow \\
\rightarrow
\end{array}
\begin{array}{c}
\boxed{fA \quad fA^\perp} \\
\boxed{fA \quad fA^\perp} \\
\begin{array}{c} fA \quad fB \\ \diagdown \quad \diagup \\ \circ \\ | \\ fA \circ fB \end{array} \\
\begin{array}{c} | \\ fA \end{array}
\end{array}$$

If Π is a pseudostructure, then the underlying linear logic pseudostructure is obtained replacing any element A of \mathcal{L} by its translation fA , dropping the indexes from ternary nodes and fusing into one single edge the edges incident to any binary node.

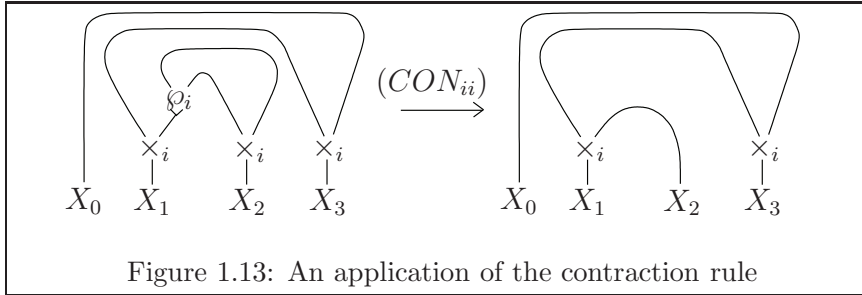
For any \mathcal{NL} (partial) structure Π , we say that Π satisfies a property P qua structure of linear logic, if $f\Pi$ satisfies P .

Substructures of \mathcal{NL} structures

Definition 1.17 A substructure \mathcal{S} of a proof structure \mathcal{R} is an induced subgraph¹⁰ of \mathcal{R} such that if a conclusion of a link L is in \mathcal{S} , then the premises of L or the other conclusion of L is in \mathcal{S} as well.

Thus a substructure \mathcal{S} of \mathcal{R} can be seen as a collection of (occurrences of) nodes of \mathcal{R} . Hence, the subset relation on such collections yields naturally a partial order on the set of substructures of a proof structure. In the same way, it makes sense to talk about the union and intersection of non-disjoint substructures. Clearly, the graphs that result from such boolean operations are still substructures.

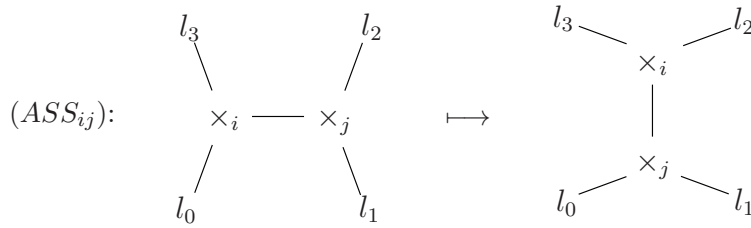
¹⁰A subgraph of a graph G is a graph whose vertex and edge sets are subsets of those of G . A subgraph H of a graph G is said to be induced if, for any pair of vertices x and y of H , xy is an edge of H if and only if xy is an edge of G .



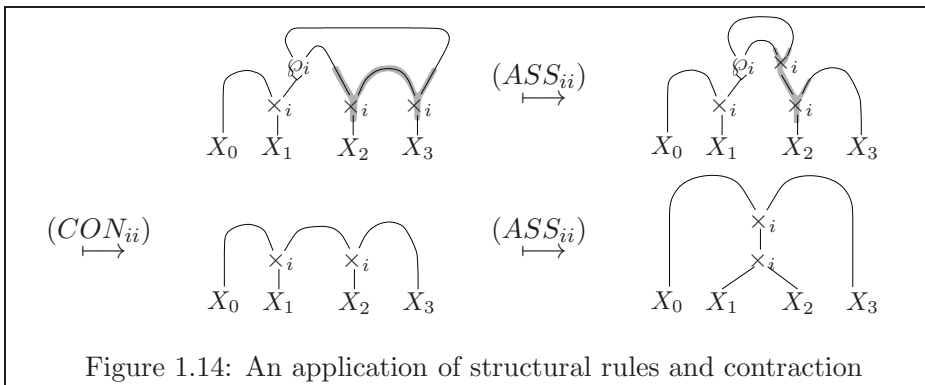
For instance, the associative structural rule for seaweeds of Figure 1.9 and the associative structural rule for trees

$$\langle H_1 \circ_i (H_2 \circ_j H_2), (H_1 \circ_i H_2) \circ_j H_3 \rangle$$

induce the rewriting rule (ASS_{ij}) defined by the following local modification of a pseudostructure:



Observe that the associativity structural rule for trees that goes in the reverse direction, i.e. $\langle (H_1 \circ_i H_2) \circ_j H_3, H_1 \circ_i (H_2 \circ_j H_2) \rangle$ induces the same associative structural rule for seaweeds and thus the same rewriting rule for pseudostructures.



Similarly, the commutative structural rule for seaweeds of Figure 1.9 and the commutative structural rule for trees $\langle H_1 \circ_i H_2, H_2 \circ_i H_1 \rangle$ induce a rewriting rule (COM_i) determined by the following local modification of a pseudostructure:

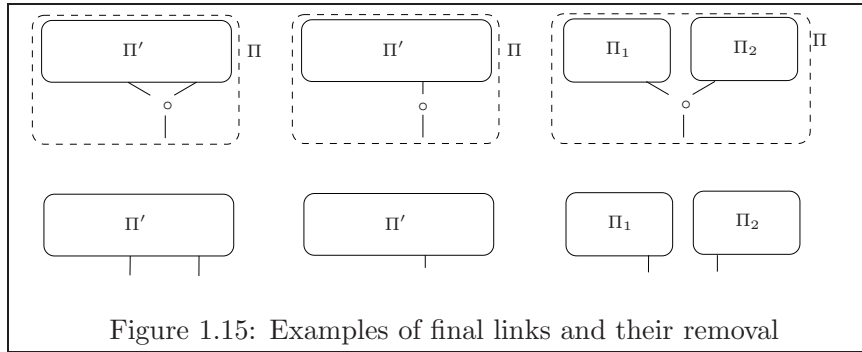
$$(COM_i): \begin{array}{ccc} \begin{array}{c} l_1 \quad l_2 \\ \diagdown \quad / \\ \times_i \\ | \\ l_0 \end{array} & \mapsto & \begin{array}{c} l_2 \quad l_1 \\ \diagdown \quad / \\ \times_i \\ | \\ l_0 \end{array} \end{array}$$

Let \mathcal{R}' be the set of rewriting rules for \mathcal{NL} pseudostructures derived from a set \mathcal{S} of structural rules. Let \mathcal{R} be the set \mathcal{R}' enriched with a set of contraction rules (CON_{ij}) , for which there are possibly some restrictions on the modalities i and j . Given any two \mathcal{NL} pseudostructures Π and Σ , say that $\Sigma \prec_{\mathcal{R}} \Pi$ if Π can be rewritten as Σ applying a rule of \mathcal{R} . Let $\leq_{\mathcal{R}}$ be the reflexive-transitive closure of $\prec_{\mathcal{R}}$.

Definition 1.20 *An \mathcal{NL} pseudostructure Π is \mathcal{R} -contractible if there is an \mathcal{NL} seaweed α such that $\alpha \leq_{\mathcal{R}} \Pi$. An \mathcal{NL} proof structure is \mathcal{R} -contractible if the underlying pseudostructure Π^- is \mathcal{R} -contractible.*

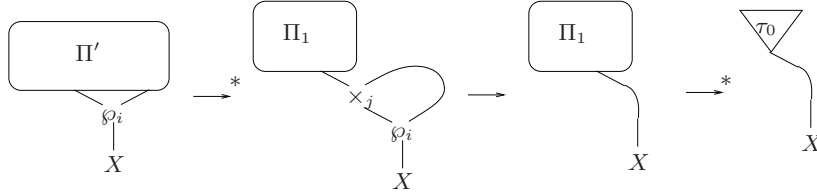
The Bridging Lemma

Definition 1.21 *For any operator \circ , a \circ -link is final in an \mathcal{NL} structure Π if its conclusion edge is connected to a unary node.*



Lemma 1.22 *Let Π be an \mathcal{R} -contractible \mathcal{NL} structure that has a final \circ - or $[\]^1$ -link L . Removing L yields an \mathcal{R} -contractible \mathcal{NL} structure.*

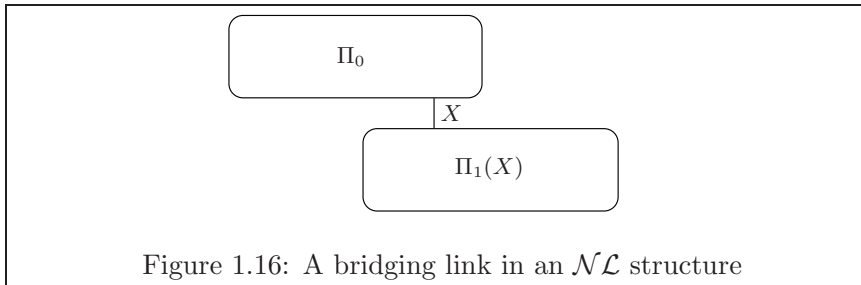
Proof. We only need to consider the case of pseudostructures. Assume that Π has a final \wp -link L and that it can be rewritten, as shown in the picture below, as a seaweed $\tau_0 \star X$ (the star on some arrow means that several rewriting rules might be involved in that step).



The central step, the contraction (CON_{ij}), can be postponed till the very end of the rewriting process, because all rewriting rules are local. This shows that Π' is \mathcal{R} -contractible. Noting that the case of a final $[\]^{-1}$ -link is entirely similar concludes the proof. \square

Definition 1.23 For any operator \circ , a \circ -link is a bridge in a \mathcal{NL} structure if the removal of its conclusion yields two disconnected components.

Any bridging \circ -link L with conclusion X (see Figure 1.16) splits a proof structure Π into a proof structure Π_0 , where L is final, and a partial proof structure $\Pi_1(X)$ with one open leaf, the edge X .

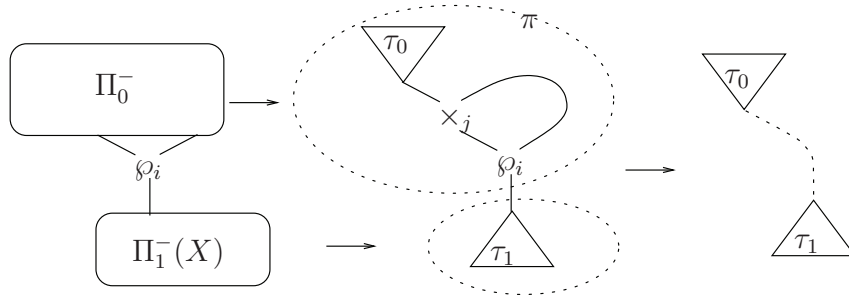


Lemma 1.24 (The Bridging Lemma) Let Π be an \mathcal{R} -contractible \mathcal{NL} structure such that:

- it has at least a non-final \wp - or $[\]^{-1}$ -link; and
- no \wp - or $[\]^{-1}$ -link is final.

Then there is a bridging \wp - or $[\]^{-1}$ -link with conclusion X that splits Π into an \mathcal{R} -contractible structure Π_0 and an \mathcal{R} -contractible partial structure $\Pi_1(X)$.

Proof. We only need to show the result for pseudostructures. Let Π be a pseudostructure and consider a rewriting process that reduces it to a seaweed. Consider the last step of the rewriting process, where a contraction c removes a link L labeled either by \wp or $[\]^{-1}$. Consider the case in which L is a \wp -link. The link L is a bridge in the pseudostructure to which the contraction c is applied, for otherwise the pseudostructure could not be rewritten as a seaweed. As pictured below, L splits this pseudostructure into a graph π and a tree τ_1 .



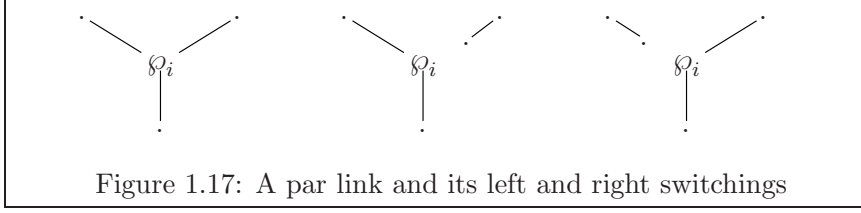
Moreover, the link L is a bridge in all the preceding pseudostructures, since no rewriting rule operates across a \wp node. Therefore, Π_0^- reduces to $\tau_0 \star X$ and $\Pi_1^-(X)$ to $\tau_1 \star X$. Thus, L splits Π into the reducible pseudostructure Π_0 and the partial pseudostructure $\Pi_1(X)$. The proof is completed observing that the case in which L is an $[\]^{-1}$ -link is entirely similar. \square

Switchings and the Danos-Regnier criteria (DR)

Definition 1.25 A switching of a par link L is the choice of one of its premises. A switching $s\Pi$ of an \mathcal{NL} structure Π is the graph that results from a switching of its \wp -links by disconnecting from each \wp -node the premise that is not selected by the switching, as indicated in Figure 1.17.

Definition 1.26 An \mathcal{NL} structure Π is said to be correct with respect to (DR), or (DR)-correct, if it satisfies the following properties:

- (DRa) each of its switchings is an acyclic graph;
- (DRc) each of its switchings is a connected graph.



The (DR) criteria are somewhat redundant.

Proposition 1.27 *Let Π be an \mathcal{NL} structure.*

- a. *If Π satisfies (DRa) and one of its switchings is connected, then Π satisfies (DRc) .*
- b. *If Π satisfies (DRc) and one of its switchings is acyclic, then Π satisfies (DRa) .*

Proof. Assume that (DRa) holds and that σ is a connected switching. Let τ be a switching of Π . Then there is a chain of switchings $\sigma = \rho_0, \dots, \rho_n = \tau$ such that, for all $i \in \{1, \dots, n\}$, ρ_{i-1} and ρ_i differ only for the switching of one par link L_i . To prove that τ is connected, it is enough to show that, for any $i \in \{1, \dots, n\}$, if ρ_{i-1} is connected, so is ρ_i . Removing the premise selected by ρ_{i-1} at L_i breaks $\rho_{i-1}\Pi$ into two components. Switching L_i as in ρ_i will either join again these two components or will create a cycle. Since the latter case violates (DRa) , $\rho_i\Pi$ must be connected. Assume now that (DRc) holds and let σ be an acyclic switching of Π . Consider a switching τ that differs from σ for the choice of premise at a \wp -node, say τ selects B at $A\wp_i B$ while σ selects A . Assume that τ contains a cycle γ . Clearly, γ must go through B because σ is acyclic. Since τ is connected, there is a path that joins A with $A \circ B$ and that does not go through B . But then also σ would be cyclic. It follows that τ , and any other switching, is acyclic. \square

In the intuitionistic case, the (DR) -criterion on connectedness is equivalent to the unicity of an output conclusion (see Theorem 2.38).

Remark 1.28 *The contraction and structural rewriting rules preserve and reflect the condition (DR) , i.e. for any \mathcal{NL} structures Σ and Π such that $\Sigma <_{\mathcal{R}} \Pi$, Σ is (DR) -correct if and only if Π is (DR) -correct.*

Since seaweeds are (DR) -correct the next result follows immediately.

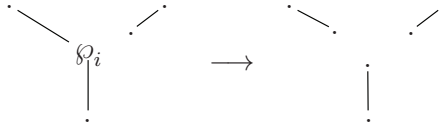
Corollary 1.29 Any \mathcal{R} -contractible structure is (DR) -correct.

Trimmings, the Danos-Regnier criteria (DR_2) , and contractibility

Lemma 1.30 Let Π be a (DR) -correct structure of linear logic. Then either it is a seaweed or it contains an elementary cycle c that goes through both premises of exactly one par link and through $n \geq 0$ tensor nodes.

Proof. Assume Π is not a seaweed. Consider a switching of Π . Because of connectedness, the two premises of any par link are joined in the switched graph by a path. Because of acyclicity, the path does not go through the conclusion of the link. Thus, for any par link, there is a path that joins the premises of the link, and goes, possibly, through some tensor node and through at most one of the premises of other par links. Choose a par link L_1 and find such a path p_1 . If p_1 contains a par link L_2 , another such path p_2 can be found. Observe that p_2 cannot go through both premises of L_1 – it lies on a switching – nor can it go through the conclusion and a premise, because otherwise the switching would be cyclic. Thus p_2 does not contain L_1 . If p_2 goes through another \wp -link, we can repeat the reasoning. But the process cannot be iterated indefinitely because proof structures are finite objects. Hence there must be a cycle c_n that goes through both premises of a par link L_n and through no other par link. \square

Consider an \mathcal{NL} structure Π . Let σ be a switching of Π and L a par link in Π . By trimming $\sigma\Pi$ at L we mean disconnecting from the \wp -node the premise of L selected by σ . If σ selects, say, the left premise the trimming process amounts to the following local modification of $\sigma\Pi$ at the switched node L :



Observe that trimming $\sigma\Pi$ at L splits the switched proof structure in a number of components. The components that contain the premises of L will be denoted by $\sigma\Pi_L$.

Definition 1.31 Let Π be an \mathcal{NL} structure and let L be a \wp -link in Π . A trimming τ of Π at L is the graph $\sigma\Pi_L$ for some switching σ of Π .

Observe that, for any (DR) -correct proof structure, a trimming of Π at a \wp -link L is a connected graph.

Definition 1.32 *Let Π be an \mathcal{NL} structure that satisfies (DR) . We say that Π satisfies (DR_2) , or is (DR_2) -correct, if and only if for any \wp -link L there is a trimming of Π at L that contains some conclusion of Π .*

The definition can be extended to cover (DR) -correct partial proof structures, requiring that for any par link there is a trimming that contains either a conclusion or an open leaf of Π .

Remark 1.33 *For any structure of linear logic, the condition (DR_2) is preserved and reflected by (CON) , (ASS) , and (COM) .*

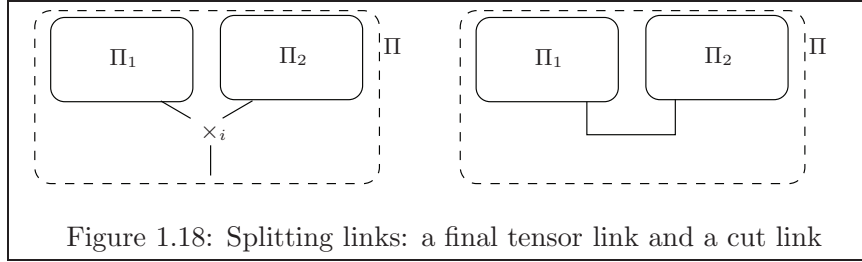
Corollary 1.34 *Any structure of linear logic that satisfies (DR_2) is contractible using (CON) , (ASS) , and (COM) .*

Proof. The proof is by induction over the complexity of the structure, defined as the number of \wp -links in the structure. If there are none, there is nothing to show. Suppose there is some \wp -link. Then by Lemma 1.30 there is an elementary cycle c that goes through both premises of exactly one par link and otherwise through edges of n \times -links, where $n \geq 0$. Since the structure is (DR_2) -correct, n cannot be zero. Therefore, possibly after applying the associativity and commutativity structural rule, applying the contraction rule yields a structure of lower complexity. Since the contraction and the structural rules preserve (DR_2) -correctness, we can apply the induction hypothesis and thus conclude the proof. \square

The Splitting Lemma

Definition 1.35 *A (DR) -correct \mathcal{NL} structure is in splitting conditions if it has no final par link, no final unary link and at least a tensor or cut link.*

Definition 1.36 *A final times- or cut-link L splits an \mathcal{NL} structure Π –or is splitting in Π – if the removal from Π of the node of L and, if it is a tensor, of its conclusion yields two disconnected \mathcal{NL} proof structures.*



Lemma 1.37 (Splitting Lemma) *Let Π be an \mathcal{NL} structure that is in splitting conditions and is (DR_2) -correct. Then there is a cut or final tensor link L that splits Π into two (DR_2) -correct substructures.*

Proof. It is enough to prove the result for the underlying structure $f\Pi$. Let n be the number of \wp -links in $f\Pi$. If $n = 0$, the result is trivial. Suppose $n \geq 1$. Because of Lemma 1.34, $f\Pi$ is contractible using the contraction rule, and the commutative and associative structural rules. By the Splitting Bridge Lemma, it splits into a proof structure Π'_0 with a final \wp -link with conclusion X and a partial proof structure $\Pi'_1(X)$, both of which must be (DR_2) -correct. Applying the induction hypothesis, one concludes that either in Π'_0 or in $\Pi'_1(X)$ there is a splitting link. This link clearly splits not only the substructure that contains it, but the whole structure. The substructures of Π that constitute the splitting inherit (DR_2) -correctness. \square

Let Π be a (DR_2) -correct \mathcal{NL} proof structure and let A be an edge of Π . Denote by $\mathcal{E}_\Pi A$ the set of (DR_2) -correct substructures of Π with A among its doors. This set is closed under intersection and union and is not empty. Indeed the proof given in [10] for the linear logic case applies; this article establishes, through a rather elaborate reasoning, a characterization of the maximum element $e_\Pi A$ of $\mathcal{E}_\Pi A$ that is used to prove the Splitting Theorem.¹¹ We can invert the reasoning here and establish the characterization using the Splitting Theorem just proved. Denote by $trim(\Pi, A)$ the set of trimmings of Π at A .

Proposition 1.38 *Let A be an edge of a (DR_2) -correct \mathcal{NL} proof structure Π . Then $e_\Pi A$ exists and*

$$e_\Pi A = \bigcap \{s : s \in trim(\Pi, A)\}.$$

¹¹The reasoning expounded in [10] has been adapted, in Chapter 4, to establish the Splitting Theorem for proof nets of the Discontinuous Lambek Calculus.

Proof. The proof is by induction over the number n of logical and cut links in Π . If $n = 0$ the result is trivial. If Π has a final par link L , removing it yields a (DR_2) -correct proof structure Π' . If A is not the conclusion of L , by induction hypothesis $e_{\Pi'}A$ exists and is the set $e_{\Pi'}A$ defined by the intersection of the trimmings of Π' at A . The conclusion of L belongs to $\bigcap\{s : s \in \text{trim}(\Pi, A)\}$ if and only if both premises of L belong to $e_{\Pi'}A$ and are different from A . Therefore $\bigcap\{s : s \in \text{trim}(\Pi, A)\}$ belongs to $\mathcal{E}_{\Pi}A$. It is the maximal element, since if another substructure \mathcal{S} would contradict its maximality, then the restriction of $e_{\Pi'}A$ to \mathcal{S} would contradict the maximality of $e_{\Pi'}A$ in $\mathcal{E}'_{\Pi}A$. If Π has a final unary link, the inductive step is immediate. To conclude, suppose that Π is in splitting conditions. The Splitting Lemma guarantees that there is a cut or final tensor link L in Π that splits it into two (DR_2) -correct substructures Π_1 and Π_2 . If L is a tensor link and A is its conclusion, the result is trivial, since $e_{\Pi}A$ is the whole Π . Suppose that A belongs to one of the substructures, say Π_1 . By induction hypothesis $e_{\Pi_1}A = \bigcap\{s : s \in \text{trim}(\Pi_1, A)\}$ and it is (DR_2) -correct. If the premise of L that belongs to Π_1 is in $e_{\Pi_1}A$, then

$$\bigcap\{s : s \in \text{trim}(\Pi, A)\} = \bigcap\{s : s \in \text{trim}(\Pi_1, A)\} \cup L \cup \Pi_2.$$

Otherwise, $\bigcap\{s : s \in \text{trim}(\Pi, A)\} = \bigcap\{s : s \in \text{trim}(\Pi_1, A)\}$. In both cases, $\bigcap\{s : s \in \text{trim}(\Pi, A)\}$ belongs to $\mathcal{E}_{\Pi}A$ and is therein the maximal element, because it inherits the maximality of $\bigcap\{s : s \in \text{trim}(\Pi_1, A)\}$ in $\mathcal{E}_{\Pi_1}A$. \square

Another consequence of the Splitting lemma is the characterization of inductive proof structures among the larger class of proof structures.

Remark 1.39 *Applying any par, bracket or antibracket rule to a proof structure Π' yields a (DR) -correct proof structure Π if and only if the original proof structure Π' is (DR) -correct. Moreover, if Π is (DR_2) -correct, then Π' is (DR_2) -correct.*

Remark 1.40 *Joining two proof structures Π_1 and Π_2 by the cut rule or any tensor rule yields a (DR) -correct proof structure Π if and only if the two proof structures Π_1 and Π_2 are (DR) -correct. Moreover, if the proof structures are (DR) -correct, then Π_1 and Π_2 are (DR_2) -correct if and only if Π is (DR_2) -correct.*

Corollary 1.41 *Any inductive \mathcal{NL} proof structure is (DR) -correct.*

Because of the Splitting Lemma we can (almost) reverse the previous result.

Corollary 1.42 *Any (DR_2) -correct \mathcal{NL} proof structure is inductive.*

Consequences of the (DR_2) criteria

Lemma 1.43 *Let Π be an \mathcal{NL} structure that satisfies (DR) and has exactly one conclusion. Then, there is a substructure \mathcal{S} of Π that is (DR) -correct, has exactly one conclusion, and the conclusion is a par type, i.e. it has the form $A\wp_i B$ for some $i \in \mathcal{N}_2$ and some suitable elements A and B of \mathcal{L} .*

Proof. The proof is by induction of the complexity n of Π , defined as the number of logical or cut links in Π . If $n = 1$ or, more generally, the conclusion of Π is a par type, the result is trivial. Assume that $n \geq 2$. There are two possible situations. In case a unary link is final, remove it and apply the induction hypothesis. Otherwise, the proof structure is in \times -splitting conditions and therefore a final cut- or tensor-link splits Π into two substructures that satisfy (DR) . At least one of them has exactly one conclusion and therefore we can conclude by applying to it the induction hypothesis. \square

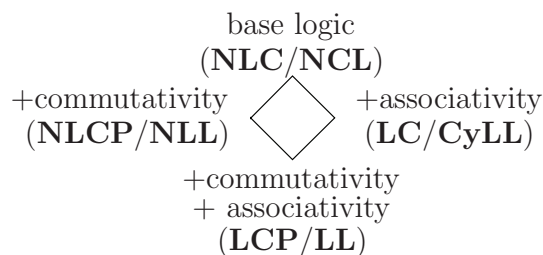
Theorem 1.44 *Let Π be an \mathcal{NL} proof structure that satisfies (DR) . Then Π satisfies (DR_2) if and only if all substructures of Π have at least two conclusions.*

Proof. To show the ‘only if’ part, assume, for contraposition, that there is a substructure \mathcal{S} of Π that has only one conclusion. Because of the previous lemma, we may assume without loss of generality that the conclusion of \mathcal{S} is a par type. We have then reached a contradiction, since no trimming at the conclusion of \mathcal{S} contains a conclusion of Π . Therefore all substructures of Π have at least two conclusions. The proof of the ‘if’ part is by induction over the complexity n of Π , defined as the number of logical or cut links in the proof structure. If $n = 0$ there is nothing to say. Otherwise, if Π has a final par-link or a final unary link, removing it yields a proof structure Π' that, by induction hypothesis, satisfies (DR_2) . Reinserting the removed link preserves this property. If Π is in splitting conditions, then a final cut- or tensor-link splits Π into two substructures. We can apply the induction hypothesis to both of them, concluding that they satisfy (DR_2) and therefore, by Remark 1.40, we can conclude that in this case too the proof structure Π satisfies (DR_2) . \square

Chapter 2

The Diamond of Basic Calculi

The Lambek Calculus **LC** ([57, 69]) is an early example of a sublinear logic. It has been studied in its commutative variant **LCP**, and in its non-associative variant **NLC** ([58, 54]).¹ For each of them there is now a classical conservative extension namely, in the given order, Cyclic Linear Logic **CyLL** ([123]), Linear Logic **LL** ([42]), and Non Associative Cyclic Linear Logic **NCL** ([28]).² The most discriminating logics (**NLC**, **NCL**) can be seen as a base logic to which the structural rules of commutativity and associativity can be added, thus yielding the following *diamond* (see Section 2.1):

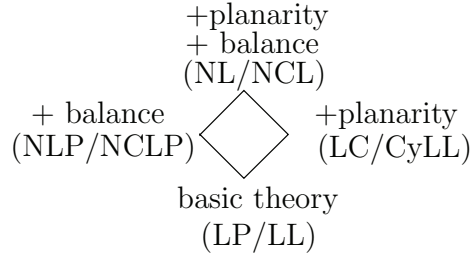


For the most lax logics, i.e. **LL** and **LCP**, a theory of proof nets is available ([42]). This theory has been adapted to the logics **CyLL** and **LC** that lack the rule of commutativity simply by adding to the correctness criterion a

¹A non-associative commutative Calculus **NLCP** has also been defined ([54]), though it does not seem to have much interest. It can be extended to a Non Associative Linear Logic **NLL**, obtained by dropping from **LL** the associative structural rule.

²Reference is intended here to the multiplicative fragments of these logics.

further clause capturing the notion of planarity ([110]). We present here another clause, *balance*, that corresponds to the absence of the structural rule of associativity. This yields the following *diamond* for proof net theory (see Sections 2.2.1 and 2.2.2):



The simplest formulation of the basic theory uses the notion of Danos-Regnier switchings ([25]). A proof structure is correct with respect to (DR) if it satisfies the following property:

(DR) each switching of the proof structure is acyclic and connected.

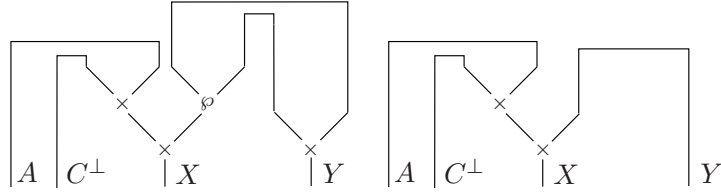
Sensitivity to order is captured by the notion of planarity of proof nets. This theory is typically stated for the cut free fragment of the logic, though a slight generalization of the notion of tree proposed in Section 2.2.3 will accommodate cut as well. A proof structure of an associative calculus sensitive to order is correct if it satisfies (DR) and moreover it enjoys the following property with respect to the given cyclic order of the conclusions:

(PL) the identity linking of the proof structure is planar.

In an alternative formulation ([65]), the order is not given a priori. Rather it is extracted from the proof net. Suppose that a proof structure Π satisfies (DR) . Disconnecting in a switching of Π the selected edge of a par link L from its conclusion splits the switched structure into two fragments. Call the trimming at L the fragment that contains L 's premises. The trimming is an (associative) seaweed and as such it represents a cyclic order. If no door of the proof structure is in between the first and the second premise of L , then we say that the premises are adjacent in the trimming. A proof structure is correct if it satisfies (DR) and enjoys moreover the following property:

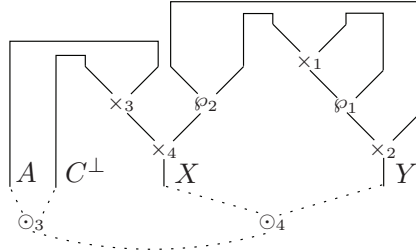
(ADJ) the premises of any par link are adjacent in any trimming at the link.

If a proof structure is correct according to $(DR)+(ADJ)$, then all of its switchings, restricted to the conclusions, define the same cyclic order. This



Observe that (BAL) is preserved and reflected by rewriting rules. Moreover, $(DR)+(BAL)$ guarantee the existence of such a minimal cycle and thus are sufficient to ensure contractibility (see Theorem 2.34).

The property (BAL) is related also to the notion of *dualizing bijective well-bracketing* ([28]) of which it is indeed a consequence in non-associative *pre-nets* à la de Groote and Lamarche. Reframing their proposal within the syntax adopted in this thesis, a pre-net can be defined to be a pair (Π, α) where Π is a proof structure, α is a seaweed and the multiset of labels of Π 's conclusions coincides with the multiset of labels of α 's unary nodes. The pair (Π, α) is represented as the graph obtained joining Π and α so that every conclusion of Π is incident to the unary node of α that bears the same label. To distinguish the two graphs, let us picture the seaweed with dotted edges and label its ternary nodes by the *context* operator \odot , as in the picture below based on the previous example (for the moment, ignore the subindexes).



The basic ingredient of de Groote and Lamarche's correctness criteria is a relation ρ defined on the ternary nodes of pre-nets. For all ternary nodes x and y , say that $x\rho y$ if x and y are the initial and final nodes of a path $\pi(x, y)$ that is the intersection of two (distinct) elementary cycles that go each through exactly one identity node. The relation ρ is said to be dualizing bijective if it establishes a bijection among the multiset of tensor nodes and the multiset of par and context nodes. The subindexes of the previous example illustrate this notion. In a correct pre-net, ρ is furthermore required to satisfy the following two conditions:⁵

⁵There is a third condition that corresponds modularly to sensitivity to order.

- *well-bracketing*: for any ternary node x, y, a , and b such that $x\rho y$ and $a\rho b$, $a \in \pi(x, y)$ if and only if $b \in \pi(x, y)$;
- (*PRT*): for any ternary node x and y if $x\rho y$ and x is the tensor node of a link L , the conclusion of L is in $\pi(x, y)$.

On the one hand, using de Groote and Lamarche’s Splitting Theorem, it is straightforward to prove that for any correct pre-net (Π, α) the proof structure Π satisfies (*DR*) and is balanced. On the other hand, if Π satisfies the latter criteria, then it can be rewritten (using contractions only) as a seaweed β . Call α the mirror image of β .⁶ Then (Π, α) is a correct pre-net, in which the relation ρ can be characterized as follows:

- a tensor and a par node are related by ρ if and only if, at some point of the derivation of β , they are removed by a contraction;
- a tensor and a context node are related by ρ if and only if, at the end of the derivation, they are related by the mirror symmetry.

This result is established, for any acyclic Π , by a straightforward induction on its complexity, and is generalized to any proof net Π observing that pre-net correctness is reflected by contractions.

The correspondence of the two theories is thus established. Observe, however, that (unsurprisingly!) correctness of a pre-net is not preserved under associative modifications of the structuring of the conclusions. Therefore the pre-net theory of [28] –unlike the theory based on the notion of balance– does not check the correctness of the proof structure *per se*, but its correctness with respect to the given structuring.

In this respect, de Groote and Lamarche’s proposal is similar to the proof net theory obtained embedding **NLC** into (commutative) Linear Logic with first order quantifiers ([76]). The embedding is obtained syntacticizing a binary relational interpretation of the Lambek Calculus ([121, 53]), which takes care of sensitivity to order, and adding one *depth* parameter. The resulting proof structure is thus frozen by the quantifiers into a non-associative non-commutative proof net the cycles of which are balanced. However, the depth parameter does not correspond exactly to non-associativity, for in a commutative context it controls only the depths of a node with respect to the root. Depth preserving restructuring is allowed.

⁶Formally, the mirror image of a tree can be defined inductively by $A^\# = A$ (if A is a trivial tree) and, for any non-trivial tree, by $(\sigma \tilde{\times} \tau)^\# = \tau^\# \tilde{\times} \sigma^\#$. The definition is extended to seaweeds setting $(\sigma \star \tau)^\# = \sigma^\# \star \tau^\#$ and observing that the definition does not depend on the choice of representation of the seaweed.

2.1 Presentation of the calculi

2.1.1 Linear Logic φ -LL

In this section we give a uniform definition, in the spirit of [69], of the logic φ -LL for any $\varphi \subseteq \{a, c\}$. If φ is equal to \emptyset , $\{a\}$ or $\{a, c\}$ the definition of the calculus φ -LL given below yields a fragment of respectively Non-associative Linear logic **NLL** ([28]), Cyclic Linear Logic **CyLL** ([123]), and Linear Logic **LL** ([42]). More precisely, they are the multiplicative, unit- and modality-free fragments of these logics, restricted to sequents that comprise at least two types.

Types and sequents of φ -LL

Definition 2.1 *The set $\mathcal{T}_{\varphi\text{-LL}}(V)$ of φ -LL types is the smallest set that contains a set V of atomic types, their negations and that is closed under the binary operations \times_{φ} and \wp_{φ} , i.e. the elements of $\mathcal{T}_{\varphi\text{-LL}}(V)$ are defined inductively in the following way:⁷*

$$\mathcal{T} = V \mid V^{\perp} \mid (\mathcal{T} \times_{\varphi} \mathcal{T}) \mid (\mathcal{T} \wp_{\varphi} \mathcal{T}).$$

A metalinguistic negation of types is defined in the following way. For every atomic type V and for any type F and G :

- $V = V^{\perp\perp}$;
- $(F\omega G)^{\perp} = G^{\perp}\omega^{\perp}F^{\perp}$ where $\omega \in \{\times_{\varphi}, \wp_{\varphi}\}$, $\times_{\varphi}^{\perp} = \wp_{\varphi}$, and $\wp_{\varphi}^{\perp} = \times_{\varphi}$.

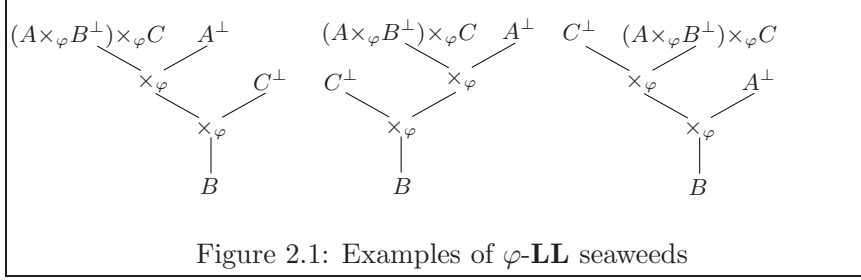
As a consequence, $F = F^{\perp\perp}$ holds for any type F .

Let $\mathcal{N} = \mathcal{N}_2 = \{\varphi\}$ and $\mathcal{L} = \mathcal{T}_{\varphi\text{-LL}}(V)$. Recall Definitions 1.1, 1.2 and 1.3 on page 19 and Definitions 1.5, 1.6 and 1.7 on page 24.

Definition 2.2 *A rigid φ -LL tree (context) is an abstract \mathcal{NL} -tree (context). A rigid φ -LL sequent (context) is an abstract \mathcal{NL} seaweed (context).*

Therefore, rigid φ -LL sequents can be pictured as in Figure 2.1, with nodes labeled by \times_{φ} and leaves labeled by φ -LL types.

⁷As usual, outer parentheses will be dropped.



Recalling the notation introduced in Definition 1.6, the leftmost seaweed of Figure 2.1, for instance, can be written abstractly in any of the following equivalent ways, where X stands for $(A \times_{\varphi} B^{\perp}) \times_{\varphi} C$:

- if we want to focus on a leaf:
 - $(C^{\perp} \tilde{\times}_{\varphi} (A^{\perp} \tilde{\times}_{\varphi} X)) \star B = [{}_{\varphi} C^{\perp}, A^{\perp} \tilde{\times}_{\varphi} X, B]$;
 - $((A^{\perp} \tilde{\times}_{\varphi} X) \tilde{\times}_{\varphi} B) \star C^{\perp} = [{}_{\varphi} A^{\perp} \tilde{\times}_{\varphi} X, B, C^{\perp}]$;
 - $(X \tilde{\times}_{\varphi} (B \tilde{\times}_{\varphi} C^{\perp})) \star A^{\perp} = [{}_{\varphi} X, B \tilde{\times}_{\varphi} C^{\perp}, A^{\perp}]$;
 - $((B \tilde{\times}_{\varphi} C^{\perp}) \tilde{\times}_{\varphi} A^{\perp}) \star X = [{}_{\varphi} B \tilde{\times}_{\varphi} C^{\perp}, A^{\perp}, X]$;
- if we prefer to see the seaweed as the \star -product of non-trivial trees:
 - $(A^{\perp} \tilde{\times}_{\varphi} X) \star (B \tilde{\times}_{\varphi} C^{\perp})$.

Rules of φ -LL

Definition 2.3 (φ -LL calculus: rigid presentation) *The rules for the φ -LL calculus of rigid sequents comprise:*

- *the identity and cut rules of Figure 2.2 (page 56);*
- *the logical rules of Figure 2.2;*
- *if $a \in \varphi$, the associative rule (ASS) of Figure 2.3 (page 57);*
- *if $c \in \varphi$, the commutative rule (COM) of Figure 2.3.*

Observe that the associative rule of Figure 2.3 can be written equivalently in any of the following forms:


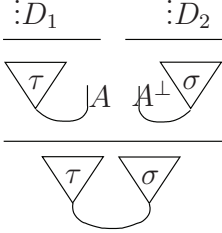
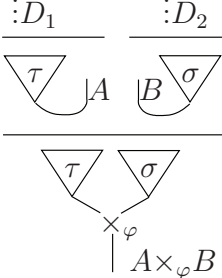
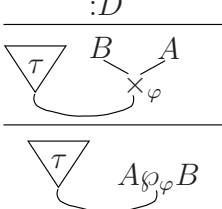
rule	abstract format	graph-theoretical format
Id	$\frac{}{A \star A^\perp}$ where A atomic	
Cut	$\frac{\frac{\vdots D_1}{\tau \star A} \quad \frac{\vdots D_2}{A^\perp \star \sigma}}{\tau \star \sigma}$	
Tensor	$\frac{\frac{\vdots D_1}{\tau \star A} \quad \frac{\vdots D_2}{B \star \sigma}}{[\varphi \tau, A \times_\varphi B, \sigma]}$	
Par	$\frac{\frac{\vdots D}{[\varphi \tau, A, B]}}{\tau \star A \wp_\varphi B}$	

Figure 2.2: φ -LL derivations: identity, cut and logical rules

rule	abstract format	graph-theoretical format
<p>(ASS) if $a \in \varphi$</p>	$\frac{\frac{\text{:D}}{[\varphi \tau_0, \tau_1, \tau_2 \times_{\varphi} \tau_3]}}{[\varphi \tau_0, \tau_1 \times_{\varphi} \tau_2, \tau_3]}}$	
<p>(COM) if $c \in \varphi$</p>	$\frac{\frac{\text{:D}}{[\varphi \tau_0, \tau_1, \tau_2]}}{[\varphi \tau_0, \tau_2, \tau_1]}}$	

Figure 2.3: φ -LL derivations: structural rules

$$\frac{\frac{\vdash D}{[\varphi \tau_0, \tau_1, \tau_2 \tilde{\times}_\varphi \tau_3]}}{[\varphi \tau_0, \tau_1 \times_\varphi \tau_2, \tau_3]} \quad \frac{\frac{\vdash D}{\tau_0 \star (\tau_1 \tilde{\times}_\varphi (\tau_2 \tilde{\times}_\varphi \tau_3))}}{\tau_0 \star ((\tau_1 \times_\varphi \tau_2) \times_\varphi \tau_3)} \quad \frac{\frac{\vdash D}{(\tau_1 \tilde{\times}_\varphi \tau_2) \star (\tau_3 \tilde{\times}_\varphi \tau_0)}}{(\tau_0 \times_\varphi \tau_1) \star (\tau_2 \times_\varphi \tau_3)}$$

Similarly, the commutative rule can be written equivalently in the following forms:

$$\frac{\frac{\vdash D}{[\varphi \tau_0, \tau_1, \tau_2]}}{[\varphi \tau_0, \tau_2, \tau_1]} \quad \frac{\frac{\vdash D}{\tau_0 \star (\tau_1 \tilde{\times}_\varphi \tau_2)}}{\tau_0 \star (\tau_2 \times_\varphi \tau_1)}$$

Definition 2.4 We say that a rigid φ -**LL** sequent α is a theorem of φ -**LL**, and we write $\vdash_{\varphi\text{-LL}} \alpha$, if and only if an appropriate application of the rules of φ -**LL** yields a derivation

$$\frac{\vdash D}{\alpha}.$$

For instance, the sequent $(C^\perp \tilde{\times}_\varphi (A^\perp \tilde{\times}_\varphi X)) \star B$, in leftmost position in Figure 2.1, can be derived in φ -**LL** if (and only if) $a, c \in \varphi$ (where $X = (A \times_\varphi B^\perp) \times_\varphi C$):

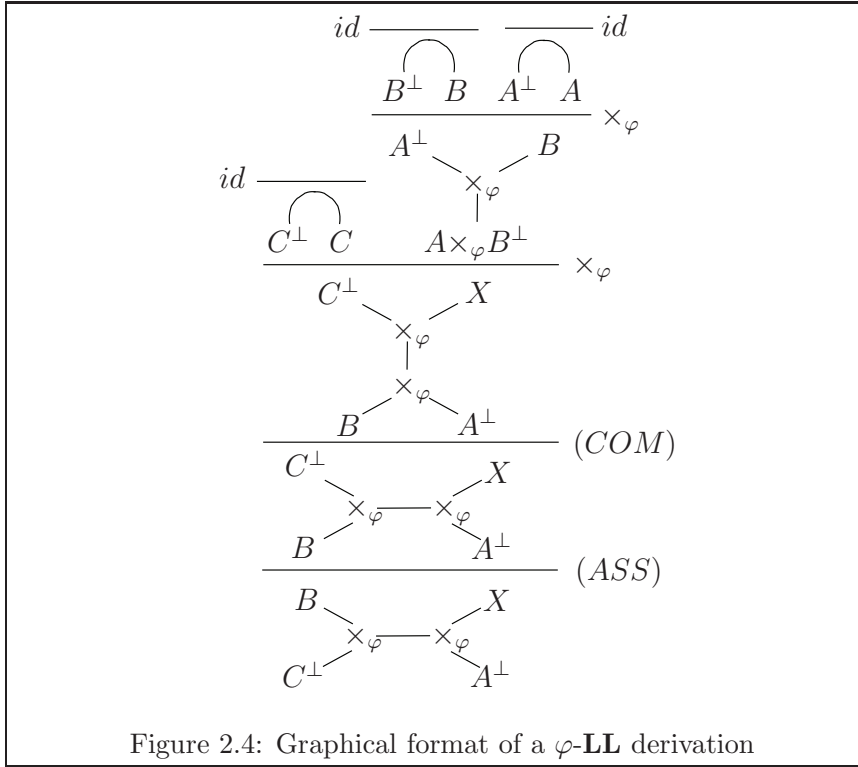
$$\frac{\frac{\frac{A^\perp \star A}{[\varphi B, A^\perp, A \times_\varphi B^\perp]} \quad \frac{B^\perp \star B}{C \star C^\perp}}{[\varphi B, A^\perp, X \times_\varphi C^\perp]}}{\frac{[\varphi B, A^\perp \tilde{\times}_\varphi X, C^\perp]}}{[\varphi B, C^\perp, A^\perp \tilde{\times}_\varphi X]}}$$

Using the graphical representation of seaweeds, this derivation can be pictured as in Figure 2.4.

A sugared presentation of φ -**LL**

On the set of rigid φ -**LL** sequents consider the equivalence relation \approx_φ generated by the transitive-reflexive closure of $<$ defined as follows:

- $H_0 \star (H_1 \circ_\varphi (H_2 \circ_\varphi H_3)) < H_0 \star ((H_1 \circ_\varphi H_2) \circ_\varphi H_3)$ if $a \in \varphi$;
- $H_0 \star (H_2 \circ_\varphi H_1) < H_0 \star (H_1 \circ_\varphi H_2)$ if $c \in \varphi$.



In Figure 2.1, the middle seaweed is \approx_φ -equivalent to the leftmost seaweed if $c \in \varphi$ and to the rightmost seaweed if $a \in \varphi$. The leftmost and rightmost seaweeds are \approx_φ -equivalents if $a, c \in \varphi$.

Definition 2.5 A sugared φ -LL sequent (context) is a φ -sugared φ -LL sequent (context), i.e. a rigid φ -LL sequent (context) identified up to the equivalence relation \approx_φ .

By a slight abuse of notation, a sugared sequent, i.e. an equivalence class $[\alpha]_{\approx_\varphi}$, will be denoted by any of its representant α .

Definition 2.6 (Sugared presentation of φ -LL) The φ -LL calculus of sugared sequents comprise the following rules of Figure 2.2 (page 56):

- the identity and cut rules;
- the logical rules.

Thus, if $\varphi = \{a\}$, a sugared φ -**LL** sequent is a list of φ -**LL** types identified up to circular permutations. If $\varphi = \{a, c\}$, a sugared φ -**LC** sequent is just a multiset of φ -**LL** types.

The Cut Elimination Property in φ -**LL**

The Cut Elimination Property holds for φ -**LL** ([42, 101, 112]), the proof being simpler in the absence of structural rules.

2.1.2 Lambek Calculus φ -**LC**

Ya ustadh, la tua fortuna è la mia fortuna[†].
I. Camera d’Affitto, *Nagib al bar della felicità*.

In this section we give a uniform definition, along the lines of [69], of the logic φ -**LC**, for any $\varphi \subseteq \{a, c\}$. If φ is equal to \emptyset , $\{a\}$ or $\{a, c\}$ the definition of the calculus φ -**LC** given below yields respectively the Non-associative Lambek Calculus **NLC** ([58]), the Lambek Calculus **LC** ([57]), and the Lambek Calculus with Permutation **LCP** ([120]).

Types and sequents of φ -**LC**

Definition 2.7 *The set $\mathcal{T}_{\varphi\text{-LC}}(V)$ of φ -**LC** types is the smallest set that contains a set V of atomic types and that is closed under the binary operations \times_{φ} , $/_{\varphi}$, and \backslash_{φ} , i.e. the elements of $\mathcal{T}_{\varphi\text{-LC}}(V)$ are defined inductively in the following way:*

$$\mathcal{T} = V \mid (\mathcal{T} \times_{\varphi} \mathcal{T}) \mid (\mathcal{T} /_{\varphi} \mathcal{T}) \mid (\mathcal{T} \backslash_{\varphi} \mathcal{T}).$$

As usual, outer parentheses will be dropped.

Let $\mathcal{N} = \mathcal{N}_2 = \{\varphi\}$ and $\mathcal{L} = \mathcal{T}_{\varphi\text{-LC}}(V)$. Recall Definitions 1.1, 1.2 and 1.3 on page 19.

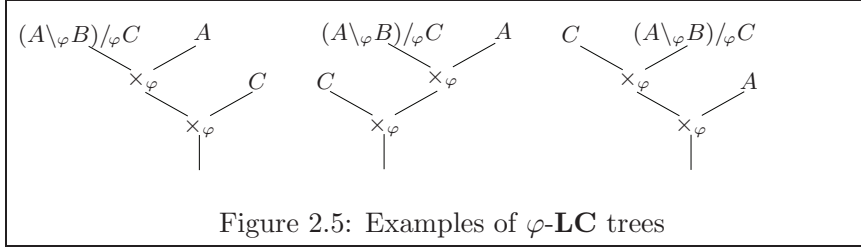
Definition 2.8 *An (abstract) rigid φ -**LC** configuration is an (abstract) \mathcal{NL} -tree.*

[†]Master, your fortune is my fortune.

Therefore abstract rigid φ -**LC** configurations are defined inductively in the following way:

$$\tau = \mathcal{L} \mid (\tau \tilde{\times}_{\varphi} \tau).$$

Rigid φ -**LC** configurations are pictured, as in Figure 2.5, as graphs with nodes labeled by \times_{φ} and leaves labeled by φ -**LC** types.



Definition 2.9 An (abstract) rigid φ -**LC** context is an (abstract) \mathcal{NL} -tree context.

Therefore, abstract rigid φ -**LC** contexts are defined inductively in the following way:

$$\sigma[] = [] \mid \sigma[] \tilde{\times}_{\varphi} \tau \mid \tau \tilde{\times}_{\varphi} \sigma[].$$

Definition 2.10 An (abstract) rigid φ -**LC** sequent $\tau \Rightarrow A$ consists of an (abstract) rigid φ -**LC** configuration τ and a φ -**LC** type A .

Rules of φ -**LC**

Definition 2.11 (Rigid presentation of φ -LC**)** The rules for the φ -**LC** calculus of rigid sequents comprise:

- the identity and cut rules of Figure 2.6;
- the logical rules of Figure 2.6;
- if $a \in \varphi$, the associative rules (ASS) of Figure 2.7;
- if $c \in \varphi$, the commutative rule (COM) of Figure 2.7.

identity and cut rules			
Id	$\frac{}{A \Rightarrow A}$ where A atomic	Cut	$\frac{\frac{\vdots D_1}{\tau \Rightarrow A} \quad \frac{\vdots D_2}{\sigma[A] \Rightarrow B}}{\sigma[\tau] \Rightarrow B}$
logical rules			
\times_φ	$\frac{\frac{\vdots D}{\sigma[A \tilde{\times}_\varphi B] \Rightarrow C}}{\sigma[A \times_\varphi B] \Rightarrow C}$	\times_φ	$\frac{\frac{\vdots D_1}{\tau_1 \Rightarrow A} \quad \frac{\vdots D_2}{\tau_2 \Rightarrow B}}{\tau_1 \tilde{\times}_\varphi \tau_2 \Rightarrow A \times_\varphi B}$
\backslash_φ	$\frac{\frac{\vdots D_1}{\tau \Rightarrow A} \quad \frac{\vdots D_2}{\sigma[B] \Rightarrow C}}{\sigma[\tau \tilde{\times}_\varphi A \backslash_\varphi B] \Rightarrow C}$	\backslash_φ	$\frac{\frac{\vdots D}{A \tilde{\times}_\varphi \tau \Rightarrow B}}{\tau \Rightarrow A \backslash_\varphi B}$
$/_\varphi$	$\frac{\frac{\vdots D_1}{\tau \Rightarrow A} \quad \frac{\vdots D_2}{\sigma[B] \Rightarrow C}}{\sigma[B /_\varphi A \tilde{\times}_\varphi \tau] \Rightarrow C}$	$/_\varphi$	$\frac{\frac{\vdots D}{\tau \tilde{\times}_\varphi A \Rightarrow B}}{\tau \Rightarrow B /_\varphi A}$

Figure 2.6: Identity, cut and logical rules of φ -LC

(ASS) if $a \in \varphi$	$\frac{\frac{\vdots D}{\sigma[(\tau_1 \tilde{\times}_\varphi \tau_2) \tilde{\times}_\varphi \tau_3] \Rightarrow C}}{\sigma[\tau_1 \times_\varphi (\tau_2 \times_\varphi \tau_3)] \Rightarrow C}$	$\frac{\frac{\vdots D}{\sigma[\tau_1 \tilde{\times}_\varphi (\tau_2 \tilde{\times}_\varphi \tau_3)] \Rightarrow C}}{\sigma[(\tau_1 \times_\varphi \tau_2) \tilde{\times}_\varphi \tau_3] \Rightarrow C}$
(COM) if $c \in \varphi$	$\frac{\frac{\vdots D}{\sigma[\tau_0 \tilde{\times}_\varphi \tau_1] \Rightarrow C}}{\sigma[\tau_1 \tilde{\times}_\varphi \tau_0] \Rightarrow C}$	

Figure 2.7: Structural rules

Definition 2.12 We say that a rigid φ -LC sequent $\tau \Rightarrow A$ is a theorem of φ -LC, and we write $\vdash_{\varphi\text{-LC}} \tau \Rightarrow A$, if and only if an appropriate application of the rules of φ -LC yields a derivation

$$\frac{\vdots D}{\tau \Rightarrow A}$$

For instance, the sequent $C \tilde{\times}_\varphi (A \tilde{\times}_\varphi X) \Rightarrow B$, can be derived in φ -LC if (and only if) $a, c \in \varphi$ (where $X = (A \backslash_\varphi B) /_\varphi C$):

$$\frac{\frac{\frac{A \Rightarrow A \quad B \Rightarrow B}{A \tilde{\times}_{\varphi} A \setminus_{\varphi} B \Rightarrow B} \quad C \Rightarrow C}{A \tilde{\times}_{\varphi} (X \tilde{\times}_{\varphi} C) \Rightarrow B}}{(A \tilde{\times}_{\varphi} X) \tilde{\times}_{\varphi} C \Rightarrow B}}{C \tilde{\times}_{\varphi} (A \tilde{\times}_{\varphi} X) \Rightarrow B}$$

The Cut Elimination Property holds for φ -**LC** (see Section 2.1.3).

A sugared presentation of φ -**LC**

On the set of rigid φ -**LC** configurations and contexts consider the smallest equivalence relation \approx_{φ} determined by the following rewriting rules:

- $(H_1 \tilde{\times}_{\varphi} H_2) \tilde{\times}_{\varphi} H_3 \xrightarrow{a} H_1 \tilde{\times}_{\varphi} (H_2 \tilde{\times}_{\varphi} H_3) \quad \text{if } a \in \varphi;$
- $H_1 \tilde{\times}_{\varphi} H_2 \xrightarrow{c} H_2 \tilde{\times}_{\varphi} H_1 \quad \text{if } c \in \varphi.$

In Figure 2.5, the middle tree is \approx_{φ} -equivalent to the leftmost tree if $c \in \varphi$ and to the rightmost tree if $a \in \varphi$. The leftmost and rightmost trees are \approx_{φ} -equivalents if $a, c \in \varphi$.

Definition 2.13 *A sugared φ -**LC** configuration (context) is a rigid φ -**LC** configuration (context) identified up to the equivalence relation \approx_{φ} .*

By a slight abuse of notation, a sugared configuration, i.e. an equivalence class $[\tau]_{\approx_{\varphi}}$, will be denoted by any of its representant τ .

Definition 2.14 *A sugared φ -**LC** sequent $\tau \Rightarrow A$ consists of a sugared φ -**LC** configuration τ and a φ -**LC** type A .*

Definition 2.15 (φ -**LC** calculus: sugared presentation) *The rules for the φ -**LC** calculus of sugared sequents comprise:*

- the identity and cut rules of Figure 2.6;
- the logical rules of Figure 2.6.

2.1.3 Embedding φ -LC into φ -LL

The intuitionistic fragment φ -ILL of φ -LL is defined restricting the rules of a two-sided presentation of φ -LL to yield sequents with only one conclusion. Alternatively, φ -ILL can be characterized as the logic obtained from the one-sided presentation of φ -LL, restricting the language to positive and negative types. We follow the latter strategy.

Intuitionistic types and sequents of φ -LL

Definition 2.16 *A φ -ILL type, i.e. an intuitionistic φ -LL type, is defined by double induction based on a set V of atomic types:*

$$\begin{aligned} \text{positive types: } & P = V \mid N \wp_{\varphi} P \mid P \wp_{\varphi} N \mid P \times_{\varphi} P; \\ \text{negative types: } & N = V^{\perp} \mid P \times_{\varphi} N \mid N \times_{\varphi} P \mid N \wp_{\varphi} N. \end{aligned}$$

Inspection of the rules of any φ -LL logic shows that theorems proved therein restricting the language to intuitionistic types have exactly one positive conclusion.

Definition 2.17 *A φ -ILL sequent, i.e. an intuitionistic φ -LL sequent, is a φ -LL sequent, where one type is positive and the others are negative.*

A cut-free derivation in φ -LL of an intuitionistic sequent is necessarily intuitionistic, i.e. it involves exclusively intuitionistic types. Since φ -LL enjoys the Cut Elimination Property, if an intuitionistic sequent can be proved at all, then it can be proved intuitionistically. More generally, a derivation of an intuitionistic statement that makes use of cut is intuitionistic if all the cut types are intuitionistic. Thus, φ -ILL enjoys the Cut Elimination Property.

The intuitionistic translation of φ -LC

Definition 2.18 *The output type A° (input type A^{\bullet}) is the translation of a φ -LC type A into a positive (negative) φ -LL type. A° is defined by induction as follows, whereas $B^{\bullet} = B^{\circ\perp}$:*

- $A^{\circ} = A$ if A is atomic;
- $A \times_{\varphi} B^{\circ} = B^{\circ} \times_{\varphi} A^{\circ}$;

- $A/\varphi B^\circ = B^\bullet \wp_\varphi A^\circ$;
- $B \setminus_\varphi A^\circ = A^\circ \wp_\varphi B^\bullet$.

Since a double negation cancels out, for any type B one has that $B^\circ = B^{\bullet\perp}$ and therefore:

- $A^\bullet = A^\perp$ if A is atomic;
- $A \times_\varphi B^\bullet = A^\bullet \wp_\varphi B^\bullet$;
- $A/\varphi B^\bullet = A^\bullet \times_\varphi B^\circ$;
- $B \setminus_\varphi A^\bullet = B^\circ \times_\varphi A^\bullet$.

The translation extends to φ -**LC** configurations.

Definition 2.19 *The intuitionistic translation of a φ -**LC** sequent $\tau \Rightarrow A$ is the φ -**LL** sequent $\tau^\bullet \star A^\circ$, where τ^\bullet is defined as follows:*

- if $\tau = B$, then $\tau^\bullet = B^\bullet$;
- if $\tau = \tau_0 \tilde{\times}_\varphi \tau_1$, then $\tau^\bullet = \tau_0^\bullet \tilde{\times}_\varphi \tau_1^\bullet$.

For instance, the intuitionistic translation of the φ -**LC** sequent

$$C \tilde{\times}_\varphi (A \tilde{\times}_\varphi (A \setminus_\varphi B) /_\varphi C) \Rightarrow B$$

is the following intuitionistic sequent (assuming that A , B , and C are atomic types):

$$\begin{aligned} & (C \tilde{\times}_\varphi (A \tilde{\times}_\varphi (A \setminus_\varphi B) /_\varphi C))^\bullet \star B^\circ &= \\ &= (C^\bullet \tilde{\times}_\varphi (A^\bullet \tilde{\times}_\varphi (A \setminus_\varphi B) /_\varphi C)^\bullet) \star B^\circ &= \\ &= C^\perp \tilde{\times}_\varphi (A^\perp \tilde{\times}_\varphi (A \times_\varphi B^\perp) \tilde{\times}_\varphi C) \star B \end{aligned}$$

Note that the rules of φ -**ILL** correspond to translations of φ -**LC** rules, so that φ -**ILL**- and φ -**LC**-derivations correspond via the translation. It follows that φ -**LC** can be faithfully embedded into φ -**LL**. In particular, cut-free φ -**ILL** derivations correspond to cut-free φ -**LC** derivations.

Theorem 2.20 *A φ -LC sequent $\tau \Rightarrow A$ is a theorem of φ -LC if and only if its intuitionistic translation $\tau^\bullet \star A^\circ$ is a theorem of φ -LL.*

Proof. The proof of the associative case (see e.g. [56]) does not make critical use of associativity and works a fortiori in the commutative case.⁸
□

Corollary 2.21 *The calculus φ -LC enjoys the Cut Elimination Property.*

2.2 Proof net theory

2.2.1 Proof net theory for φ -LL

This section is dedicated to the notion of proof nets and correctness criteria for φ -LL. For the procedural criteria, we draw on [108] and [74]. For the declarative criteria, we draw on [10, 65] and, for the notion of balance, on the author's work ([36]).

Notion of φ -LL proof structure

Definition 2.22 *A φ -LL (partial) proof structure is an \mathcal{NL} (partial) proof structure, where $\mathcal{N} = \mathcal{N}_2 = \{\varphi\}$ and $\mathcal{L} = \mathcal{T}_{\varphi\text{-LL}}(V)$.*

Therefore, recalling Definition 1.9, a φ -LL partial proof structure is a connected graph such that:

- its edges are labeled by elements of $\mathcal{T}_{\varphi\text{-LL}}(V)$;
- its ternary nodes are labeled by elements of \times_φ or \wp_φ ; incident edges are ordered cyclically; one of them is specified to be the conclusion, the left (right) premise is the edge that immediately follows (precedes) the conclusion in clockwise fashion;
- its binary nodes are labeled by ID and CUT (or more simply, represented by a horizontal line); ID nodes have two conclusions, CUT nodes have two premises;

⁸For an explicit proof of the non-commutative non-associative case see [28].



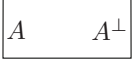

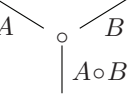
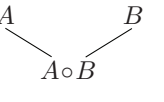
	link	premises	conclusions	compact link
id		none	A and A^\perp	
cut		A and A^\perp	none	
logical		A and B	$A \circ B$	

Figure 2.8: Links of a φ -LL proof structure

- unary nodes are not labeled;
- there are no other nodes;
- any edge is a premise of at most one node and the conclusion of at most one node.

For any node label l , a node labeled by l together with its incident edges is called an l -link. The labels of a link are related in the way explained in the first four columns of Figure 2.8.

Proof nets as classes of equivalent derivations

Proof nets, i.e. proof structures that can be sequentialized, have been introduced in [42] to identify those proofs that differ in inessential ways. To be more precise, the following two definitions are needed.

Definition 2.23 *Two φ -LL derivations \mathcal{D} and \mathcal{D}' are \equiv -equivalent if there exists a sequence of φ -LL derivations $\mathcal{D} = \mathcal{D}_1, \mathcal{D}_2, \dots, \mathcal{D}_n = \mathcal{D}'$ such that, for any $i \in \{1, \dots, n-1\}$, \mathcal{D}_i and \mathcal{D}_{i+1} differ only in a permutation of two consecutive inferences.*

Definition 2.24 *The proof structure associated to a φ -LL derivation \mathcal{D} is the inductive proof structure (\mathcal{D}) obtained following the rules given in Figure 2.9.*

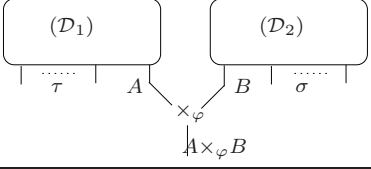
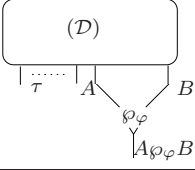
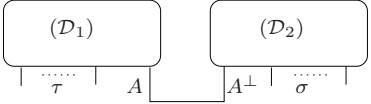
derivation	proof structure
$\frac{}{A \star A^\perp}$	$A \overline{\quad} A^\perp$
$\frac{\frac{\vdots D_1}{\tau \star A} \quad \frac{\vdots D_2}{B \star \sigma}}{[\varphi \tau, A \times_\varphi B, \sigma]}$	
$\frac{\frac{\vdots D}{[\varphi \tau, A, B]}}{\tau \star A \wp_\varphi B}$	
$\frac{\frac{\vdots D_1}{\tau \star A} \quad \frac{\vdots D_2}{A^\perp \star \sigma}}{\tau \star \sigma}$	

Figure 2.9: Inductive definition of the φ -**LL** proof structure (\mathcal{D})

Clearly if two derivations are \equiv -equivalent then their associated proof nets coincide. The inverse result is not trivial. The following proof, given in [10] for **LL**, applies without any modification to φ -**LL**, for any $\varphi \subseteq \{a, c\}$, provided that we use the sugared presentation of the calculi.

Theorem 2.25 *Let \mathcal{D} and \mathcal{D}' be φ -**LL** derivations of the same sequent. If $(\mathcal{D}) = (\mathcal{D}')$ then $\mathcal{D} \equiv \mathcal{D}'$.*

Proof. By induction over the complexity n of $(\mathcal{D}) = (\mathcal{D}')$. If $n = 0$ there is nothing to show. Assume $n \geq 1$. Let I_0 be the last inference rule in \mathcal{D} and let $\tau \star A$ be the theorem proved by \mathcal{D} . Let I'_0 be the inference rule in \mathcal{D}' that introduces A . Let I'_1, \dots, I'_k be the inferences between I'_0 and the last inference I'_k of \mathcal{D}' . If $k = 0$, then it is enough to apply the induction hypothesis to the derivations of the premise sequents of I_0 and I'_0 . If $k \geq 1$, it can be shown that in \mathcal{D}' the rule I'_0 can be permuted below I'_1 to obtain a derivation \mathcal{D}'' that is equivalent to \mathcal{D}' . By induction hypothesis $\mathcal{D}'' \equiv \mathcal{D}'$ and thus by transitivity \mathcal{D}' is equivalent to \mathcal{D} . The cases of I'_0 being a par rule or both I'_0 and I'_1 being binary rules, are immediate. If I'_0 is a tensor rule and I'_1 is a par rule, then the derivations have the following form:

$$\mathcal{D}: \frac{\frac{\dot{:}D_0}{\tau_0 \star A_0} \quad \frac{\dot{:}D_1}{A_1 \star \tau_1}}{[\tau_0, A_0 \times_{\varphi} A_1, \tau_1]} \quad \mathcal{D}': \frac{\frac{\frac{\dot{:}D'_0}{\tau'_0 \star A_0} \quad \frac{\dot{:}D'_1}{A_1 \star \tau'_1}}{[\tau'_0, A_0 \times_{\varphi} A_1, \tau'_1]}}{[\dots, C_0 \wp_{\varphi} C_1, \dots]}}{\vdots}$$

where I'_1 introduces $C_0 \wp_{\varphi} C_1$ in \mathcal{D}' . Both C_0 and C_1 are in $(\mathcal{D}'_0) \cup (\mathcal{D}'_1)$. The rule I'_0 can be permuted below I'_1 if there is $j \in \{0, 1\}$ such that C_0 and C_1 are in (\mathcal{D}'_j) . This is the case because on the one hand there is $j \in \{0, 1\}$ such that C_0 and C_1 are in (\mathcal{D}_j) , and $\mathcal{D}_0 \cap \mathcal{D}_1$ is empty; and on the other hand, if C_i is in (\mathcal{D}'_h) (for any i, h in $\{0, 1\}$) then C_i is connected to A_h by a path that does not go through $A_0 \times_{\varphi} A_1$ and thus C_i is in (\mathcal{D}_h) . \square

The notion of pseudostructure

Definition 2.26 A φ -**LL** pseudostructure is an \mathcal{NL} pseudostructure, where $\mathcal{N} = \mathcal{N}_2 = \{\varphi\}$ and $\mathcal{L} = \mathcal{T}_{\varphi\text{-LL}}(V)$.

Therefore, recalling Definition 1.14, a φ -**LL** pseudostructure is a connected graph such that:

- its ternary nodes are labeled either by \times_{φ} or by \wp_{φ} ; incident edges are ordered cyclically; in the case of the \wp_{φ} nodes, one of the edges is specified to be the conclusion (and denoted by a tail on the edge);
- unary nodes are labeled by φ -**LL** types;
- there are no other nodes;
- edges are not labeled.

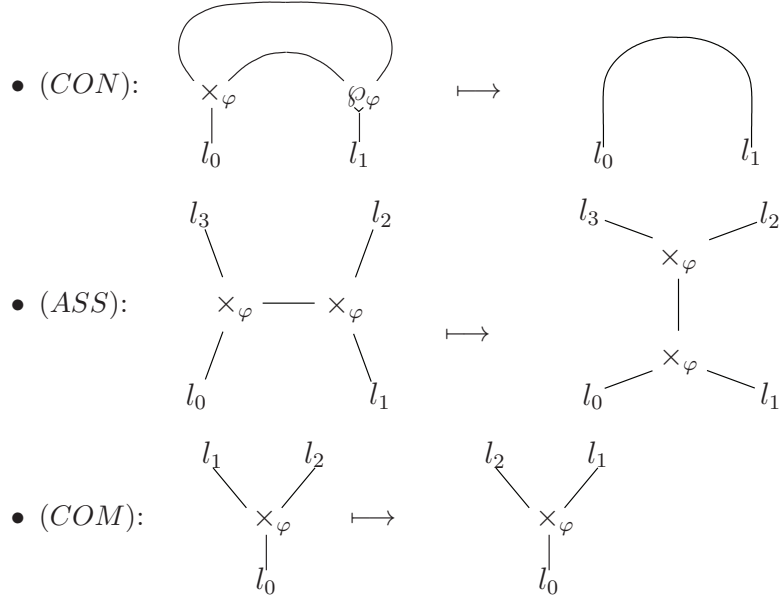
A \wp_{φ} -link is a node labeled by \wp_{φ} together with its incident edges.

We denote by Π^- the pseudostructure associated to a φ -**LL** proof structure Π , according to the instructions given on page 35.

Procedural correctness criteria

Definition 2.27 A φ -**LL** proof structure Π is φ -contractible and φ -rewrites as a φ -**LL** sequent α if the underlying pseudostructure Π^- rewrites as α

using the contraction rule (CON) and the structural rules (ASS) if $a \in \varphi$, and (COM) if $c \in \varphi$:



Theorem 2.28 (Adequacy) *If \mathcal{D} is a φ -LL derivation of a sequent α , then (\mathcal{D}) rewrites as α .*

Proof. By induction over the length n of \mathcal{D} . If $n = 0$, i.e. if α is an axiom, there is nothing to prove. If $n \geq 1$ consider the last rule r in \mathcal{D} . If r is an instance of the par rule

$$\frac{\frac{\frac{\cdot}{\mathcal{D}_0}}{[\varphi \tau, A, B]}}{\tau \star A \wp_\varphi B}}$$

then by induction hypothesis $(\mathcal{D}_0)^-$ rewrites as $[\varphi \tau, A, B]$. The same rewriting rules applied to $(\mathcal{D})^-$ yield the graph obtained joining in $[\varphi \tau, A, B]$ the leaves labeled by A and B by a par link with conclusion $A \wp_\varphi B$. A further contraction yields the seaweed $\tau \star A \wp_\varphi B$. If the last rule used in \mathcal{D} is an instance of the tensor rule

$$\frac{\frac{\frac{\cdot}{\mathcal{D}_0}}{\tau_0 \star A_0} \quad \frac{\frac{\cdot}{\mathcal{D}_1}}{A_1 \star \tau_1}}{[\varphi \tau_0, A_0 \times_\varphi A_1, \tau_1]}}$$

then by induction hypothesis $(\mathcal{D}_i)^-$, following a list ρ_i of rewriting rules, rewrites as $\tau_i \star A_i$ for $i \in \{0, 1\}$. Following the lists ρ_1 and ρ_2 , $(\mathcal{D})^-$ yields $[\varphi \tau_0, A_0 \times_{\varphi} A_1, \tau_1]$. If \mathcal{D} consists of a derivation \mathcal{D}' followed by an instance of a structural rule s , apply the induction hypothesis to $(\mathcal{D}')^-$ and rewrite the resulting sequent applying the rewriting rule that corresponds to s . The proof is completed observing that the case of cut is analogous to the tensor case (with the minor difference that no further contraction is needed). \square

Theorem 2.29 (Sequentialization) *For any φ -LL proof structure Π that rewrites as a sequent α , there is a φ -LL derivation \mathcal{D} of α such that $(\mathcal{D}) = \Pi$.*

Proof. By induction over the complexity n of Π . If $n = 0$ there is nothing to show. Suppose Π has a final \wp_{φ} -node L , the conclusion of which is labeled by X . By Lemma 1.22, the removal of L yields a contractible proof structure Π' . By induction hypothesis, Π' can be sequentialized as a derivation \mathcal{D}' . Completing \mathcal{D}' with a \wp_{φ} -rule that introduces X as the main formula yields a derivation \mathcal{D} of α such that $\Pi = (\mathcal{D})$. If Π has no final \wp_{φ} -node, then, by the Bridge Splitting Lemma 1.24, there is a bridging \wp_{φ} -link that decomposes Π into a proof structure Π_0 and a partial proof structure $\Pi_1(X)$ such that Π_0^- and $\Pi_1(X)^-$ are contractible. Note that $\Pi_1(X)$ can be turned into a proof structure Π_1 simply adding an identity link with conclusions X and X^{\perp} . $\Pi_1(X)$ and Π_1 coincide qua unlabeled graphs and therefore also Π_1^- is contractible. By induction hypothesis, there are derivations \mathcal{D}_0 and \mathcal{D}_1 such that $(\mathcal{D}_0) = \Pi_0$ and $(\mathcal{D}_1) = \Pi_1$. Replacing in the derivation \mathcal{D}_1 the axiom $X \star X^{\perp}$ with the derivation \mathcal{D}_0 yields a derivation \mathcal{D} of α such that $(\mathcal{D}) = \Pi$. \square

Invariants for the rewriting rules

It has already been observed that the contraction rule and the associative and commutative rules preserve and reflect property (DR) and the stricter property (DR_2) .

Since the associative and commutative rewriting rules are reversible, indeed idempotent, any declarative correctness criterion must be, intuitively speaking, a maximal invariant for these rules.

For commutativity, the solution was presented in [65] by R. Maieli.

Definition 2.30 Let Π be an \mathcal{NL} structure that satisfies (DR_2) . Let τ be a trimming of Π at a φ -link L . Consider a door D of Π that belongs to τ . The paths that join pairwise D and the premises of L meet at $center_\tau(L, D)$, a node called the center in τ of L and D .

Call π_1 , π_2 , and π_D the paths in τ that join $center_\tau(L, D)$ with, respectively, the first and the second premise of L and the door D .

Definition 2.31 We say that D is not between the premises of L in τ if the paths π_1 , π_2 , and π_D are in this cyclic order when moving counterclockwise around $center_\tau(L, D)$. We say that the premises of L are adjacent in τ if no door of Π that belongs to τ is between the premises of L .

Definition 2.32 A φ -LL structure Π that is (DR_2) -correct satisfies condition (ADJ) if for any trimming τ at any par link L the premises of L are adjacent in τ .

Clearly, the contraction and associative rewriting rules preserve and reflect this property. The commutative rewriting rule destroys it.

Observe that associativity changes the number of tensor nodes on elementary cycles, while it does not affect the number of par nodes. Any contraction that affects an elementary cycle removes a pair made of a par node and a tensor node. Commutativity does not affect the number of par/tensor nodes on cycles. All this suggests the following characterization.

Definition 2.33 A cycle in a φ -LL structure is balanced if it contains the same number of par nodes and tensor nodes. A φ -LL structure is balanced, or satisfies condition (BAL) , if each of its elementary cycles is balanced.

Clearly, any seaweed is, trivially, a balanced pseudostructure. Therefore, any φ -LL proof structure that can be contracted without using the associativity rule is balanced.

Declarative correctness criteria

Theorem 2.34 Let Π be a (DR_2) -correct proof structure of φ -LL. Suppose moreover that Π satisfies condition (BAL) if $a \notin \varphi$ and (ADJ) if $c \notin \varphi$. Then Π φ -rewrites as a seaweed α and the restriction of any switching of Π to its conclusions is \approx_φ -equivalent to α .

Proof. By induction over the number of ternary nodes of Π . If there is none, there is nothing to prove. If Π has a final par link L with hypothesis A and B , consider the proof structure Π' obtained from Π by removing the link L . Since Π' inherits from Π the correctness conditions, by induction hypothesis it φ -rewrites as a seaweed α' and any switching of Π' is φ -equivalent to α' . Joining the leaves A and B of α' with the link L yields a pseudostructure ρ with one cycle that goes through one par node –the central node of L – and through $n \geq 1$ tensor nodes. If $a \notin \varphi$, then $n = 1$, for otherwise Π could not have been balanced since all rewriting rules, except the associative one, reflect balance. If $a \in \varphi$, the associative rule can be applied so as to decrease the number of tensor nodes in the cycle, till there is just one. Hence, in any case, we can assume that Π φ -rewrites as a pseudostructure ρ that contains one cycle γ' and that γ' goes through exactly one par node and one tensor node. If $c \notin \varphi$, ρ satisfies the condition (ADJ) , since this property is preserved by all rewriting rules except the commutative one. If $c \in \varphi$, the commutative rewriting rule can be applied. In any case we may assume that ρ' , and therefore Π , φ -rewrites as a pseudostructure ρ'' that contains only a cycle γ'' and that γ'' is a planar cycle that goes through exactly one tensor node and one par node. Hence applying a contraction rule the pseudostructure ρ'' , and therefore Π , can be rewritten as a seaweed φ -equivalent to α . If Π has no final par then, qua proof net of Linear Logic, it splits into two substructures joined by a link L . Each of the substructures inherits the correctness conditions and thus, by induction hypothesis, contracts to a seaweed. Joining them by L yields a seaweed to which Π contracts. \square

The main result follows immediately from the above theorems.

Theorem 2.35 *Let Π be a φ -LL proof structure, and α a φ -LL sequent. The following facts are equivalent:*

- *there is a φ -LL derivation \mathcal{D} of α such that $(\mathcal{D}) = \Pi$ and any switching of Π is φ -equivalent to α ;*
- *Π φ -rewrites as α ;*
- *Π satisfies conditions (DR_2) and, moreover,*
 - *(ADJ) if $c \notin \varphi$;*
 - *(BAL) if $a \notin \varphi$.*

2.2.2 Proof net theory for φ -LC

In this section we transfer the results of the previous section to φ -LC via the embedding of Section 2.1.3. Therefore, the only original results are those for the non-associative versions of the Lambek Calculus.

We have recalled in Section 2.1.3 that the intuitionistic fragment of φ -LL can be obtained from φ -LL restricting the language to positive and negative types. Thus the proof nets for the former logic are simply the proof net of the latter logic with labelling restricted to such types. Also, intuitionistic φ -LL derivations correspond, via the notion of polarity, to φ -LC derivations. Hence φ -LC proof nets are essentially intuitionistic φ -LL proof nets. However, one would like the proof nets of φ -LC to make reference only to the very same logic. This is achieved labelling the proof nets not with positive and negative types, but directly with the polar Lambek types. Labelling of the links is changed according to the following two rules:

- if a label is a positive φ -LL type p , replace it with P° , where P is the unique φ -LC type such that $P^\circ = p$;
- if a label is a negative φ -LL type n , replace it with N^\bullet , where N is the unique φ -LC type N such that $N^\bullet = n$.

The repertoire of links is reported in Figure 2.10, where if s is a polarity, \bar{s} denotes the opposite polarity.

identity	$A^p \boxed{} A^{\bar{p}}$	cut	$A^p \boxed{} A^{\bar{p}}$
\times_{φ}^\bullet	$\begin{array}{c} A^\bullet \backslash \quad / B^\bullet \\ \quad \quad \quad \rho_{\varphi} \\ \quad \quad \quad \\ \quad \quad \quad A \times_{\varphi} B^\bullet \end{array}$	\times_{φ}°	$\begin{array}{c} B^\circ \backslash \quad / A^\circ \\ \quad \quad \quad \times_{\varphi} \\ \quad \quad \quad \\ \quad \quad \quad A \times_{\varphi} B^\circ \end{array}$
$\backslash_{\varphi}^\bullet$	$\begin{array}{c} A^\circ \backslash \quad / B^\bullet \\ \quad \quad \quad \times_{\varphi} \\ \quad \quad \quad \\ \quad \quad \quad A \backslash_{\varphi} B^\bullet \end{array}$	$\backslash_{\varphi}^\circ$	$\begin{array}{c} B^\circ \backslash \quad / A^\bullet \\ \quad \quad \quad \rho_{\varphi} \\ \quad \quad \quad \\ \quad \quad \quad A \backslash_{\varphi} B^\circ \end{array}$
$/_{\varphi}^\bullet$	$\begin{array}{c} A^\bullet \backslash \quad / B^\circ \\ \quad \quad \quad \times_{\varphi} \\ \quad \quad \quad \\ \quad \quad \quad A /_{\varphi} B^\bullet \end{array}$	$/_{\varphi}^\circ$	$\begin{array}{c} B^\bullet \backslash \quad / A^\circ \\ \quad \quad \quad \rho_{\varphi} \\ \quad \quad \quad \\ \quad \quad \quad A /_{\varphi} B^\circ \end{array}$

Figure 2.10: Links for φ -LC

Observe that there are three tensor links, namely the \times_{φ}° , $\backslash_{\varphi}^\bullet$, and $/_{\varphi}^\bullet$ links and three par links, namely the \times_{φ}^\bullet , $\backslash_{\varphi}^\circ$, and $/_{\varphi}^\circ$ links. The correctness conditions carry over without any change. As for the structuring α_{Π} of the doors that is implicitly contained in a correct proof structure Π , observe

that it is always possible to focus on the unique output door, call it A° . Therefore any such structuring α can be expressed as $\tau^\bullet \star A^\circ$ for a suitable φ -**LC** configuration τ of the input doors.

Therefore, Theorem 2.35 can be restated for φ -**LC** as follows.

Theorem 2.36 *Let Π be a φ -**LC** proof structure, and let $\tau \Rightarrow A$ be a φ -**LC** sequent. The following facts are equivalent:*

- *there is a φ -**LC** derivation \mathcal{D} of $\tau \Rightarrow A$ such that $(\mathcal{D}) = \Pi$ and any switching of Π is φ -equivalent to $\tau^\bullet \star A^\circ$;*
- *Π φ -rewrites as $\tau^\bullet \star A^\circ$;*
- *Π satisfies conditions (DR_2) and, moreover,*
 - *(ADJ) if $c \notin \varphi$;*
 - *(BAL) if $a \notin \varphi$.*

Condition (DR) , and therefore (DR_2) , can be stated in an equivalent way for any φ -**LC** proof structure, changing the clause (DRc) on connectedness for the clause (DRu) on uniqueness of the output door:

- (DRa) each of its switchings is an acyclic graph;
- (DRu) the proof structure has a unique output door.

This claim, common lore in the literature, is motivated by a more general result.

Definition 2.37 *A component of a graph is a subgraph which is connected and is maximal, with respect to inclusion, among connected subgraphs.*

Theorem 2.38 *Let \mathcal{R} be a φ -**LC** proof-structure with no cyclic switching. The following facts are equivalent:*

- *\mathcal{R} has n output doors;*
- *a switching of \mathcal{R} has n components;*
- *every switching of \mathcal{R} has n components.*

Proof. The proof is by induction on the number n of logical and cut links of \mathcal{R} . If $n = 0$, then \mathcal{R} has only doors that are conclusions of identity links and the result is trivial. If $n \geq 1$ there are two cases:

- Suppose that \mathcal{R} has a door that is the conclusion of a par link L and let \mathcal{R}' be the proof structure obtained from \mathcal{R} removing L . Then \mathcal{R} and \mathcal{R}' have the same number n of output doors. By induction hypothesis, one/every switching of \mathcal{R}' has n components. The switchings of \mathcal{R} can be obtained from the switchings of \mathcal{R}' by adding an edge connecting one of the premises of L with the conclusion of L . This operation does not affect the number of components and thus one/every switching of \mathcal{R} has n components.
- Suppose that \mathcal{R} has a door that is the conclusion of a tensor link L . If \mathcal{R} has n output doors, the proof structure \mathcal{R}' obtained from \mathcal{R} removing L has $n + 1$ output doors and, by induction hypothesis, one/any switching s of \mathcal{R}' has $n + 1$ components. The premises of L must belong to different components of every s , otherwise \mathcal{R} would have cyclic switchings. Since the switchings of \mathcal{R} can be obtained reinserting the link L in the switchings of \mathcal{R}' , any switching of \mathcal{R} will have n components.

□

2.2.3 Planar proof nets

In this section we further study proof nets for the two non-commutative variants of φ -LL, i.e. the cases in which $c \notin \varphi$. We discuss the notion of planarity of a proof net ([110]), formalizing the idea of ordering the leaves of a forest of trees, by ordering their roots and, eventually, *grafting* some of them with some *cut tree*.

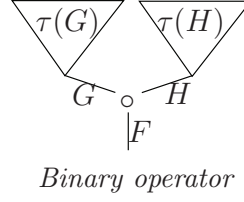
Following the definition of type tree, we can define the notion of *cut tree*.

Definition 2.39 *A type tree $\tau(F)$ is a graph defined as follows by induction on the complexity of a φ -LL type F :*

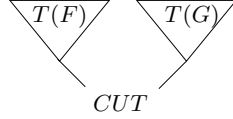
- *if F is atomic, then $\tau(F)$ is the graph that consists of one edge labeled by F and two unary unlabeled nodes;*

$\begin{array}{c} | \\ F \\ | \end{array}$
Trivial type tree

- if $F = G \circ H$ for some $\circ \in \{\times_\varphi, \wp_\varphi\}$, then $\tau(F)$ is the graph obtained from $\tau(G)$ and $\tau(H)$ joining, around a ternary node labeled by \circ , the edges labeled by G and H with a further edge labeled by F ; moreover, the edges labeled by F , G , and H must be clockwise in this cyclic order around the mentioned node.



Definition 2.40 A cut tree $\tau_{cut}(F, G)$ consists of a cut link that joins, in the given order, two type trees $\tau(F)$ and $\tau(G)$, where F and G are any φ -LL types such that $F = G^\perp$. Graphically:



Clearly, any φ -LL proof structure can be thought of as a collection of type and cut trees, the leaves of which are joined pairwise by identity links.

Definition 2.41 An identity linking on a set of leaves $\mathcal{F} = \{F_1, \dots, F_n\}$ is a collection of identity links the sets of conclusions of which constitute a partition of \mathcal{F} .

When a proof structure contains no cut link, it is easy to see how ordering the trees that constitute the proof structure yields an ordering of their leaves. If there is some cut link, some extra work has to be done.

To simplify the wording of the definitions and results of this section, we add to any cut link a further edge, the *conclusion* of the link, labeled by the extra-symbol \underline{C} .

Observe that for any type or cut tree T we can consider on the set of its edges two relations, whereby an edge *immediately precedes* another edge and edge is *immediately dominated* by another edge. The relations are defined as follows:

- $A \leq_{ip}^{\{T\}} B$ if $\langle A, B \rangle$ is the list of premises of a link of T ;
- $A \leq_{id}^{\{T\}} B$ if A is a premise of a link of T with conclusion B .

The only edge that is not immediately dominated by any other label is the root of the tree. The edges that do not immediately dominate any edge are the leaves of the tree.

Definition 2.42 *The set of φ -LL grafted trees is the smallest set \mathcal{G} such that:*

- for every type and cut tree T , $(\{T\}, \langle_{ip}^{\{T\}}, \langle_{id}^{\{T\}} \rangle)$ is in \mathcal{G} ;
- for any edge label A , B , and D of a grafted tree $(\mathcal{T}, \langle_{ip}^{\mathcal{T}}, \langle_{id}^{\mathcal{T}} \rangle)$ such that $A \langle_{ip}^{\mathcal{T}} B$ and $A, B \leq_{id}^{\mathcal{T}} D$ and for any cut tree S with root \underline{C} , $(\mathcal{T}' = \mathcal{T} \cup \{S\}, \langle_{ip}^{\mathcal{T}'}, \langle_{id}^{\mathcal{T}'} \rangle)$ is in \mathcal{G} , where:

$$\begin{aligned} \langle_{ip}^{\mathcal{T}'} &= (\langle_{ip}^{\mathcal{T}} \setminus \{ \langle A, B \rangle \}) \cup \langle_{ip}^{\{S\}} \cup \{ \langle A, \underline{C} \rangle, \langle \underline{C}, B \rangle \}, \text{ and} \\ \langle_{id}^{\mathcal{T}'} &= \langle_{id}^{\mathcal{T}} \cup \langle_{id}^{\{S\}} \cup \{ \langle \underline{C}, D \rangle \}. \end{aligned}$$

A simple example of a grafted tree is $(\{T(A \varphi B), S\}, \langle)$ where S is the cut-link with premises E and E^\perp and the partial orders are given by:

- $\langle_{ip} = \{ \langle A, \underline{C} \rangle, \langle \underline{C}, B \rangle \}$, and
- $\langle_{id} = \{ \langle A, A \varphi B \rangle, \langle \underline{C}, A \varphi B \rangle, \langle B, A \varphi B \rangle, \langle E, \underline{C} \rangle \}$.

It can be represented graphically as in Figure 2.11.

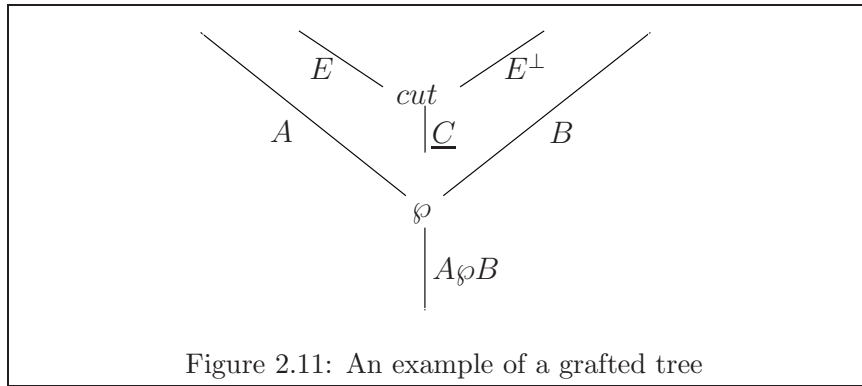


Figure 2.11: An example of a grafted tree

Let \mathcal{S} be a multiset of grafted trees and let R be a strict order on the set of roots of \mathcal{S} . The leaves of \mathcal{S} are then partially ordered in a natural way. Consider the following relations:

- the dominance relation \leq_d^S defined as the reflexive-transitive closure of $\bigcup_{T \in \mathcal{S}} <_{id}^T$;
- the generalization $<_{ip}^S$ of the immediate precedence relation, defined as $R \cup \bigcup_{T \in \mathcal{S}} <_{ip}^T$;
- the precedence relation $<_p^S$ defined as follows: $A < B$ exactly when there are C and D such that $A <_d^S C$, $C <_{ip}^S D$, and $B <_d^S D$.

The restriction of $<_p^S$ to the set of leaves of \mathcal{S} is a total strict order. Let $\langle F_1, \dots, F_n \rangle$ be the sequence of leaves, ordered according to $<_p^S$.

Definition 2.43 *An identity linking I on the sequence $\langle F_1, \dots, F_n \rangle$ of leaves is \mathcal{S} -plane if there are no links in I with conclusions $\{F_{i_1}, F_{i_2}\}$ and $\{F_{i_3}, F_{i_4}\}$ such that $i_1 < i_3 < i_2 < i_4$.*

Note that the above property is invariant under circular permutations. Thus it can be asserted also of the sequence $\langle F_1, \dots, F_n \rangle$ identified up to permutations induced by circular permutations of the sequence of grafted trees.

Observe that any φ -**LL** proof structure Π that is represented on the plane with non-intersecting trees consists of a sequence \mathcal{S} of grafted trees and an identity linking on their leaves.

Definition 2.44 *A φ -**LL** proof structure Π is \mathcal{S} -plane if the identity linking on the set of leaves of \mathcal{S} is \mathcal{S} -plane.*

Theorem 2.45 *Let Π be a (DR_2) -correct proof structure that satisfies (BAL) if $a \notin \varphi$. Π can be represented in the plane by a sequence \mathcal{S} of grafted trees and an \mathcal{S} -plane identity linking if and only if Π satisfies condition (ADJ) .*

Proof. Clearly, if Π is \mathcal{S} -plane then it satisfies (ADJ) . Conversely, if it satisfies (ADJ) then Π can be sequentialized as a φ -**LL** derivation \mathcal{D} . The following Proposition shows that the proof structure (\mathcal{D}) associated to \mathcal{D} can be represented in the plane by a sequence \mathcal{S} of grafted trees and an \mathcal{S} -plane identity linking. \square

Recall that any sequent α establishes –on the multiset $\{F_1, \dots, F_n\}$ of its types– a cyclic order, that can be represented by any cyclic permutation of a sequence $\langle F_{i_1}, \dots, F_{i_n} \rangle$.

Proposition 2.46 *Let \mathcal{D} be a φ -LL derivation of a sequent α . Then it is possible to represent (\mathcal{D}) in the plane by a sequence \mathcal{S} of grafted trees and an \mathcal{S} -plane identity linking. Moreover, \mathcal{S} can be chosen so that the restriction of $\langle _ \rangle_p^{\mathcal{S}}$ to $\{F_1, \dots, F_n\}$ is the strict order $\langle F_{i_1}, \dots, F_{i_n} \rangle$.*

Proof. The proof is by induction over the length of the derivation \mathcal{D} . If the derivation is just an identity axiom, there is nothing to show. For the inductive step, the following table gives –case by case– the definition of the sequence \mathcal{S} . The first column indicates the last rule of the derivation. The second column indicates the sequence(s) of grafted trees that exist by induction hypothesis. Here χ , χ_1 and χ_2 stand for sequences of cut trees and γ (respectively δ) stands for a sequence of grafted trees that contains, in the given order, the types that constitute the tree Γ (Δ). Finally, the third and fourth columns contain the definition of \mathcal{S} and explain how to obtain the tree T by grafting.

induction step	ind. hypothesis	\mathcal{S}	to obtain T
$\frac{\frac{\vdots \mathcal{D}'}{[\varphi \Gamma, A, B]}}{\Gamma \star A \wp_{\varphi} B}$	$\gamma, \tau(A), \chi, \tau(B)$	γ, T	graft χ in $\tau(A \wp_{\varphi} B)$ between A and B
$\frac{\frac{\vdots \mathcal{D}'}{\Gamma \star A} \quad \frac{\vdots \mathcal{D}''}{B \star \Delta}}{[\varphi \Gamma, A \times_{\varphi} B, \Delta]}$	$\gamma, \tau(A), \chi_1$ and $\chi_2, \tau(B), \delta$	γ, T, δ	graft χ_1, χ_2 in $\tau(A \times_{\varphi} B)$ between A and B
$\frac{\frac{\vdots \mathcal{D}'}{\Gamma \star A} \quad \frac{\vdots \mathcal{D}''}{A^{\perp} \star \Delta}}{\Gamma \star \Delta}$	$\gamma, \tau(A), \chi_1$ and $\chi_2, \tau(A^{\perp}), \delta$	γ, T, δ	graft χ_1, χ_2 in $\tau(A, A^{\perp})$ between A and A^{\perp}

Clearly, the planarity of (\mathcal{D}) follows from the induction hypothesis. □

Chapter 3

Combining and Constraining Basic Calculi

The Lambek Calculus **LC** is complete both with respect to models based on ternary accessibility relations ([34]) and to powerset residuated semigroup models ([15]).

On the one hand, the former result has been extended in [53] to any multimodal Lambek Calculus that comprises unary and binary modalities and in which the structural rules are weak Sahlqvist ([53]).¹ On the other hand, the latter result has been shown in [52] to hold for the Lambek Calculus generalized by the addition of any family of n -ary operators ($n \geq 1$) together with the respective residuated operators.

Thus, model-theoretically, the behaviour of unary and binary operators does not differ. Their similarity is corroborated by the procedural theory of proof nets ([74, 108]) that is based on one contraction rule for any tensor-par pair and one for any bracket-antibracket pair. But, from a declarative point of view, the situation seems to be different.

First of all, there is no general declarative theory of proof nets. For this reason, we consider in this chapter the most restrictive cases of multimodality –namely those systems where structural rules make reference to only one modality– and we propose a correctness criterion that captures the notion of *endomodality*, i.e. the lack of intermodal communication. Any system with structural rules that mix modalities demands for a relaxation of such a criterion.

¹A structural rule is weak Sahlqvist if, loosely speaking, it does not result in any form of weakening or contraction. Beside [53], a formal definition is found, e.g., in [69].

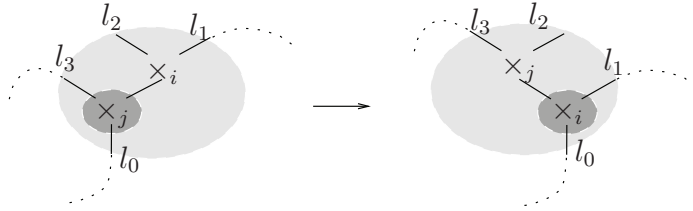
Secondly, the notion of Danos-Regnier switching can be extended easily to any n -ary residuated operators. A switching of a proof structure is the graph that results from a switching of its \wp -links by disconnecting from each \wp -node L all the premises that are not selected at L by the switching. But, for $n = 1$, this means that there is no difference between an antibracket link L and its switching. This suggests that, in a declarative correctness theory, links of unary operators must be treated in a different way.² I propose a solution based on a translation into an endomodal calculus.

This chapter has the following structure. Section 3.1 presents the theory of endomodal calculi, first for some classical logics and then for their intuitionistic fragments.

The lack of mixed rules is captured by the following condition on (DR_2) -correct proof structures:

for any modality i , any trimming τ at any \wp_i -link L , and any conclusion D of the proof structure that belongs to τ the paths that join pairwise D and the premises of L meet at a node labeled by a tensor of modality i .

Clearly, this property does not hold if there is a mixed associative rule, as exemplified by the following picture:



If all the modalities of an endomodal calculus are non-associative, the property of *balance*, relativized to each modality, entails the condition defined above. In particular, this observation characterizes proof nets for Moortgat and Morrill's biheaded calculus ([71]) devised to account for mismatches in head/dependent and functor/argument structures.³

²See also the discussion in [76], that leaves as an open question the possibility of finding an embedding translation of unary connectives into the Lambek Calculus with Permutation and first order quantification.

³The idea of using a bimodal calculus to account for the mismatches in head/dependent and functor/argument structures has been taken up in some multimodal versions of Combinatory Categorical Grammar, see e.g. [6].

Section 3.2 enriches with brackets the calculi of the previous section. In particular, Sections 3.2.2 and 3.2.3 –that are based on the joint publication [37] with G. Morrill– build upon and correct Versmissen’s proposal of mimicking brackets with two extra symbols ([122]). The basic idea –that the extra symbols must be *matched properly*– comes from my own work on pregroups with brackets ([35]) reported in the final chapter of this thesis. Section 3.2.4 contains the proposal of mimicking brackets with one extra symbol, say P , and one extra modality, say j :

- $s([\]A) = P \times_j sA$;
- $s([\]^{-1}A) = P^\perp \wp_j sA$.

This proposal has the advantage of working also in commutative environments. In particular, it can be seen as a first step toward a correctness theory, where commutativity is licensed locally by a unary modality.

3.1 Endomodal Calculi

3.1.1 Linear Logic Φ -LL

In this section we study multimodal linear logics, obtained combining versions of linear logic with/without commutativity and/or associativity, where no structural rule makes reference to more than one modality. The logics are defined and a theory of proof nets is proposed. The central notion is the property (ENDO), which corresponds to the absence of interaction among modalities. The properties of balance and adjacency are relativized to each modality.

Let Φ be a possibly empty subset of $\{a_i, c_i : i \in \mathcal{N}_2\}$, where \mathcal{N}_2 is the set of indexes of the (binary) modalities.

Rules for Φ -LL

Definition 3.1 *The set $\mathcal{T}_{\Phi\text{-LL}}(V)$ of Φ -LL types is the smallest set that contains a set V of atomic types, their negations and that is closed, for any $i \in \mathcal{N}_2$, under the binary operations \times_i and \wp_i , i.e. the elements of $\mathcal{T}_{\Phi\text{-LL}}(V)$ are defined inductively in the following way:*

$$\mathcal{T} = V \mid V^\perp \mid \mathcal{T} \times_i \mathcal{T} \mid \mathcal{T} \wp_i \mathcal{T}.$$

A metalinguistic negation of types is defined in the following way. For all atomic types A , for any type F and G , and for any $i \in \mathcal{N}_2$:

- $A = A^{\perp\perp}$;
- $(F \times_i G)^\perp = G^\perp \wp_i F^\perp$;
- $(F \wp_i G)^\perp = G^\perp \times_i F^\perp$.

As a consequence, $F = F^{\perp\perp}$ holds for any type F .

Definition 3.2 A Φ -**LL** sequent is an \mathcal{NL} seaweed where $\mathcal{N} = \mathcal{N}_2$ and $\mathcal{L} = \mathcal{T}_{\Phi\text{-LL}}(V)$.

Definition 3.3 The Φ -**LL** calculus of sequents comprises the following rules, defined in Figure 3.1:

- the identity and cut rules;
- the logical rules;
- the structural rules determined by Φ .

Identity and Cut rules			
Id	$\frac{}{A \star A^\perp}$ where A atomic	Cut	$\frac{\frac{\vdash D_1}{\Gamma \star A} \quad \frac{\vdash D_2}{A^\perp \star \Delta}}{\Gamma \star \Delta}$
Logical Rules (for any $i \in \mathcal{N}_2$)			
Tensor	$\frac{\frac{\vdash D_1}{\Gamma \star A} \quad \frac{\vdash D_2}{B \star \Delta}}{[i \Gamma, A \times_i B, \Delta]}$	Par	$\frac{\vdash D}{[i \Gamma, A, B]}$ $\Gamma \star A \wp_i B$
Structural rules			
(ASS_i) if $a_i \in \Phi$	$\frac{(\tau_0 \times_i \tau_1) \star (\tau_2 \times_i \tau_3)}{(\tau_1 \times_i \tau_2) \star (\tau_3 \times_i \tau_4)}$		
(COM_i) if $c_i \in \Phi$	$\frac{[i \tau_0, \tau_1, \tau_2]}{[i \tau_0, \tau_2, \tau_1]}$		

Figure 3.1: Rules of Φ -**LL**

The usual proof of the Cut Elimination Property applies to Φ -**LL** too.

Proof Net Theory for Φ -LL

Definition 3.4 A Φ -LL structure –a pseudostructure/(partial) proof structure– is an \mathcal{NL} structure –a pseudostructure/(partial) proof structure– where $\mathcal{N} = \mathcal{N}_2$, and $\mathcal{L} = \mathcal{T}_{\Phi\text{-LL}}(V)$.

Recall Definition 2.30 of the center of a par link L and a door in a trimming at L .

Definition 3.5 Let Π be a (DR_2) -correct structure of Φ -LL. For any modality $i \in \mathcal{N}_2$, we say that Π is i -endocentric, or that it satisfies property $(i\text{-ENDO})$, if for any \wp_i -link L in Π , any trimming τ of Π at L and any conclusion D in τ , the center in τ of L and D is of modality i . We say that Π is endocentric, or that it satisfies property $(ENDO)$, if Π is i -endocentric for any modality $i \in \mathcal{N}_2$.

Lemma 3.6 Let Π be an \mathcal{NL} structure that satisfies (DR_2) , and $(i\text{-ENDO})$ for some $i \in \mathcal{N}_2$. Removal of a final \wp -link and splitting by a cut- or final \times -link preserve correctness with respect to these conditions.

Proof. Preservation of (DR_2) has been already established by Remarks 1.39 and 1.40. Let Π' be the structure obtained by removing from Π a final \wp -link L with conclusion D . To show that Π' is $(i\text{-ENDO})$ -correct, consider a \wp_i -link M with conclusion E and a trimming τ' of Π' at M . Let C be a conclusion of Π' that belongs to the trimming τ' . If C is a premise of L , extend τ' to a trimming τ of Π , by selecting C at L . Then, $center_{\tau'}(M, C) = center_{\tau}(L, D)$ must be of modality i . If C is not among the premises of L , select at L any of its premises. Then $center_{\tau'}(M, C) = center_{\tau}(M, C)$. Therefore, in any case, the center in τ' of M and C has modality i . A similar reasoning proves preservation of $(i\text{-ENDO})$ under splitting. Indeed, let a cut- or final \times -link split Π into two structures Π_0 and Π_1 . Consider a \wp_i -link M with conclusion E in, say, Π_0 . Let τ_0 be a trimming of Π_0 at M and let C be a conclusion of Π_0 that belongs to τ_0 . If C is not a premise of the splitting link, let $D = C$. If it is, call D any conclusion of Π_1 that is not a premise of the splitting link. Observe that such a conclusion exists, because Π_1 satisfies (DR_2) . Extend in any way τ_0 to a trimming of Π at M . Then $center_{\tau_0}(M, C)$ coincides with $center_{\tau}(M, D)$ and has therefore modality i . \square

The correctness conditions (ADJ) and (BAL) have to be relativized to each modality $i \in \mathcal{N}_2$.

Definition 3.7 Let Π be an \mathcal{NL} structure that is (DR_2) -correct. We say that Π satisfies condition (ADJ_i) if for any trimming τ at any \wp_i -link L the premises of L are adjacent in τ .

Definition 3.8 Let Π be an \mathcal{NL} structure. A cycle in Π is i -balanced if it contains the same number of \wp_i -nodes and \times_i -nodes. Π is i -balanced, or satisfies condition (BAL_i) , if each of its elementary cycles is i -balanced. Π is balanced if it is i -balanced for any modality $i \in \mathcal{N}_2$.

Lemma 3.9 Let Π be a Φ -LL structure that satisfies (DR_2) and $(j$ -ENDO) for any modality j of a subset $J \subseteq \mathcal{N}_2$. Assume moreover that, for all $j \in J$, (BAL_j) holds if and only if $a_j \notin \Phi$ and (ADJ_j) holds if and only if $c_j \notin \Phi$. Then one of the following facts hold:

1. Π is a seaweed;
2. for some $j \in J$, there is a cycle γ that goes through both premises of a \wp_j -link L and through other nodes that are all tensors of the same modality j ; moreover, if $a_j \notin \Phi$, γ goes through exactly one tensor; if $c_j \notin \Phi$, then for any trimming τ at L , no conclusion is in between the premises of L in τ ;
3. for some $i \in \mathcal{N}_2 \setminus J$ there is a cycle γ that goes through both premises of a \wp_i -link and through other nodes that are all tensors (not necessarily of the same modality).

Proof. The proof is by induction over the complexity of Π , defined as its number of logical links. If $n = 0$ there is nothing to show. If, for some modality h , Π has a final \wp_h -link L , removing it yields a structure Π' that, because of the previous lemma, inherits from Π all the correctness conditions. By induction hypothesis, either Π' , and therefore Π , contains a cycle with the sought for properties, or Π' is a seaweed. In the latter case, reinserting the final \wp -link L yields a cycle γ each tensor node of which is, in the only possible trimming of Π , the center of L and a conclusion of Π . If $h \notin J$, some tensor belongs to the cycle because of (DR_2) , and no further check is needed. If $h \in J$, all these tensors must be of modality j because Π is j -endocentric. Moreover, if $a_j \notin \Phi$, then by (BAL_j) the cycle γ has to contain exactly one \times_j -node; if $c_j \notin \Phi$, no conclusion of Π can be in between the premises of L in the only possible trimming at L . If Π is in splitting conditions, then it splits into two substructures Π_0 and Π_1 that are connected by a splitting link L . The previous lemma ensures that both Π_0 and Π_1 inherit the correctness conditions from Π . By induction hypothesis,

either some of the substructures, and therefore Π , contains the sought for cycle γ , or both substructures, and therefore Π , are seaweeds. \square

Observe that in the previous proof, the condition (ENDO) is needed only in one step (when reinserting a removed final \wp -link) and only if there is some modality that is associative. Indeed, if none is associative, (BAL_i) holds for any i . Then for any $i \neq h$ there cannot be a \times_i in γ and for $i = h$ there must be exactly one.

Corollary 3.10 *Let Π be an Φ -LL proof structure that satisfies (DR_2) and $(ENDO)$. Assume moreover that, for all $i \in \mathcal{N}_2$, (BAL_i) holds if and only if $a_i \notin \Phi$ and (ADJ_i) holds if and only if $c_i \notin \Phi$. Then one of the following facts hold:*

1. Π is a seaweed; or
2. Π contains a cycle γ that goes through both premises of a \wp link of some modality i and through other nodes that are all tensors of the same modality i ; moreover, if $a_i \notin \Phi$, γ goes through exactly one tensor; if $c_i \notin \Phi$, then for any trimming τ at L , no conclusion is in between the premises of L in τ .

Lemma 3.11 *Let Π be a Φ -LL pseudostructure. The following facts hold:*

- contractions, non-mixed associative rules, and commutative rules preserve and reflect (DR) , $(CONCL_{\geq 2})$, $(i-ENDO)$ for any modality i ;
- for all modalities $i \neq j$, (ASS_i) preserves and reflects (BAL_j) ;
- for all modalities $i \neq j$, (COM_i) preserves and reflects (ADJ_j) .

Corollary 3.12 *Let Π be a Φ -LL proof structure. The following facts are equivalent:*

- Π is Φ -LL correct, i.e. it satisfies (DR_2) , $(ENDO)$, (BAL_i) for any modality i such that $a_i \notin \Phi$, and (ADJ_i) for any modality i such that $c_i \notin \Phi$;
- Π can be rewritten as a seaweed using contractions and the rewriting rules corresponding to the structural rules of Φ -LL.

Corollary 3.13 *A Φ -LL proof structure is sequentializable in Φ -LL if and only if it is Φ -LL correct.*

3.1.2 Lambek Calculi Φ -LC

In this section we study multimodal Lambek calculi, obtained by combining versions of the Lambek Calculus with/without commutativity and/or associativity, where no structural rule makes reference to more than one modality. The calculi are embedded, via the usual notion of polarity, in the linear logics of the previous section and therefore inherit the proof net theory.

Rules for Φ -LC

Definition 3.14 *The set $\mathcal{T}_{\Phi\text{-LC}}(V)$ of Φ -LC types is the smallest set that contains a set V of atomic types and that is closed, for any $i \in \mathcal{N}_2$, under the binary operations \times_i , $/_i$, and \setminus_i , i.e. the elements of $\mathcal{T}_{\Phi\text{-LC}}(V)$ are defined inductively in the following way (where $i \in \mathcal{N}_2$):*

$$\mathcal{T} = V \mid (\mathcal{T} \times_i \mathcal{T}) \mid (\mathcal{T} /_i \mathcal{T}) \mid (\mathcal{T} \setminus_i \mathcal{T}).$$

As usual, outer parentheses will be dropped.

Let $\mathcal{N} = \mathcal{N}_2$ and $\mathcal{L} = \mathcal{T}_{\Phi\text{-LC}}(V)$. Recall Definitions 1.1, 1.2 and 1.3 on page 19.

Definition 3.15 *An (abstract) Φ -LC configuration is an (abstract) \mathcal{NL} -tree.*

Therefore abstract Φ -LC configurations are defined inductively in the following way (where $i \in \mathcal{N}_2$):

$$\tau = \mathcal{L} \mid (\tau \tilde{\times}_i \tau).$$

Definition 3.16 *An (abstract) Φ -LC context is an (abstract) \mathcal{NL} -tree context.*

Therefore, abstract Φ -LC contexts are defined inductively in the following way (where $i \in \mathcal{N}_2$):

$$\sigma[] = [] \mid \sigma[] \tilde{\times}_i \tau \mid \tau \tilde{\times}_i \sigma[].$$

Definition 3.17 *An (abstract) Φ -LC sequent $\tau \Rightarrow A$ consists of an (abstract) Φ -LC configuration τ and a Φ -LC type A .*

Definition 3.18 The calculus of Φ -LC sequents comprises the following rules, defined in Figure 3.2:

- the identity and cut rules;
- the logical rules;
- the structural rules that belong to Φ .

Identity and cut rules			
Id	$\frac{}{A \Rightarrow A}$ where A atomic	Cut	$\frac{\frac{\dot{D}_1}{\tau \Rightarrow A} \quad \frac{\dot{D}_2}{\sigma[A] \Rightarrow B}}{\sigma[\tau] \Rightarrow B}$
Logical rules (for any $i \in \mathcal{N}_2$)			
\times_i	$\frac{\frac{\dot{D}}{\sigma[A \tilde{\times}_i B] \Rightarrow C}}{\sigma[A \times_i B] \Rightarrow C}$	\times_i	$\frac{\frac{\dot{D}_1}{\tau_1 \Rightarrow A} \quad \frac{\dot{D}_2}{\tau_2 \Rightarrow B}}{\tau_1 \tilde{\times}_i \tau_2 \Rightarrow A \times_i B}$
\setminus_i	$\frac{\frac{\dot{D}_1}{\tau \Rightarrow A} \quad \frac{\dot{D}_2}{\sigma[B] \Rightarrow C}}{\sigma[\tau \tilde{\times}_i A \setminus_i B] \Rightarrow C}$	\setminus_i	$\frac{\frac{\dot{D}}{A \times_i \tau \Rightarrow B}}{\tau \Rightarrow A \setminus_i B}$
$/_i$	$\frac{\frac{\dot{D}_1}{\tau \Rightarrow A} \quad \frac{\dot{D}_2}{\sigma[B] \Rightarrow C}}{\sigma[B /_i A \tilde{\times}_i \tau] \Rightarrow C}$	$/_i$	$\frac{\frac{\dot{D}}{\tau \tilde{\times}_i A \Rightarrow B}}{\tau \Rightarrow B /_i A}$
Structural rules			
(ASS_i)	$\frac{\frac{\dot{D}}{\sigma[(\tau_1 \tilde{\times}_\varphi \tau_2) \tilde{\times}_\varphi \tau_3] \Rightarrow C}}{\sigma[\tau_1 \tilde{\times}_\varphi (\tau_2 \tilde{\times}_\varphi \tau_3)] \Rightarrow C}$		$\frac{\frac{\dot{D}}{\sigma[\tau_1 \tilde{\times}_\varphi (\tau_2 \tilde{\times}_\varphi \tau_3)] \Rightarrow C}}{\sigma[(\tau_1 \tilde{\times}_\varphi \tau_2) \tilde{\times}_\varphi \tau_3] \Rightarrow C}$
(COM_i)	$\frac{\frac{\dot{D}}{\sigma[\tau_0 \tilde{\times}_\varphi \tau_1] \Rightarrow C}}{\sigma[\tau_1 \tilde{\times}_\varphi \tau_0] \Rightarrow C}$		

Figure 3.2: Rules of Φ -LC calculus

Definition 3.19 We say that a Φ -LC sequent $\tau \Rightarrow A$ is a theorem of Φ -LC, and we write $\vdash_{\Phi\text{-LC}} \tau \Rightarrow A$, if and only if an appropriate application of the rules of Φ -LC yields a derivation

$$\frac{\dot{D}}{\tau \Rightarrow A}$$

Embedding Φ -LC into Φ -LL

The same reasoning expounded in the previous chapter on the embedding of φ -LC into φ -LL can be applied for the new calculi, with only very minor modifications in the definitions. The proofs are essentially the same and will therefore be omitted.

Definition 3.20 A Φ -ILL type, i.e. an intuitionistic Φ -LL type, is defined by double induction based on a set V of atomic types (where $i \in \mathcal{N}_2$):

$$\begin{aligned} \text{positive types: } & P = V \mid N \wp_i P \mid P \wp_i N \mid P \times_i P; \\ \text{negative types: } & N = V^\perp \mid P \times_i N \mid N \times_i P \mid N \wp_i N. \end{aligned}$$

Definition 3.21 A Φ -ILL sequent, i.e. an intuitionistic Φ -LL sequent, is a Φ -LL sequent, where one type is positive and the others are negative.

Definition 3.22 The output type A° (input type A^\bullet) is the translation of a Φ -LC type A into a positive (negative) Φ -LL type. A° is defined by induction as follows, whereas $B^\bullet = B^{\circ\perp}$:

- $A^\circ = A$ if A is atomic;
- $A \times_i B^\circ = B^\circ \times_i A^\circ$;
- $A /_i B^\circ = B^\bullet \wp_i A^\circ$;
- $B \setminus_i A^\circ = A^\circ \wp_i B^\bullet$.

Since a double negation cancels out, for any type B one has that $B^\circ = B^{\bullet\perp}$ and therefore:

- $A^\bullet = A^\perp$ if A is atomic;
- $A \times_i B^\bullet = A^\bullet \wp_i B^\circ$;
- $A /_i B^\bullet = A^\circ \times_i B^\circ$;
- $B \setminus_i A^\bullet = B^\circ \times_i A^\circ$.

The translation extends to Φ -LC configurations.

Definition 3.23 *The intuitionistic translation of a Φ -LC sequent $\tau \Rightarrow A$ is the sequent $\tau^\bullet \star A^\circ$, where τ^\bullet is defined as follows:*

- if $\tau = B$, then $\tau^\bullet = B^\bullet$;
- if $\tau = \tau_0 \tilde{\times}_i \tau_1$, then $\tau^\bullet = \tau_0^\bullet \tilde{\times}_i \tau_1^\bullet$.

Theorem 3.24 *A Φ -LC sequent $\tau \Rightarrow A$ is a theorem of Φ -LC if and only if its intuitionistic translation $\tau^\bullet \star A^\circ$ is a theorem of Φ -LL.*

Corollary 3.25 *The calculus Φ -LC enjoys the Cut Elimination Property.*

Proof net theory for Φ -LC

Because of the embedding established in the previous section, the proof nets of Φ -LC are essentially the proof nets of Φ -LL with labelling restricted to positive and negative types. To obtain proof nets that make reference only to the theory of Φ -LC, the labelling of the links is changed according to the following two rules:

- if a label is a positive Φ -LL type p , replace it with P° , where P is the unique Φ -LC type such that $P^\circ = p$;
- if a label is a negative Φ -LL type n , replace it with N^\bullet , where N is the unique Φ -LC type N such that $N^\bullet = n$.

The repertoire of links is reported in Figure 3.3, where i is any modality and if s is a polarity, \bar{s} denotes the opposite polarity.

The correctness conditions carry over from the previous section without any change. As for the structuring α_Π of the doors that is implicitly contained in a correct proof structure Π , observe that it is always possible to focus on the unique output door, call it A° . Therefore any such structuring α_Π can be expressed as $\tau^\bullet \star A^\circ$ for a suitable Φ -LC configuration τ of the input doors.

Therefore, the sequentialization results of the previous section can be restated for Φ -LC as follows.

Theorem 3.26 *Let Π be a Φ -LC proof structure and let $\tau \Rightarrow A$ be a Φ -LC sequent. The following facts are equivalent:*

identity	$A^p \boxed{\phantom{A^p \phantom{A^{\bar{p}}}}} A^{\bar{p}}$	cut	$A^p \boxed{\phantom{A^p \phantom{A^{\bar{p}}}}} A^{\bar{p}}$
\times_i^\bullet	$\begin{array}{c} A^\bullet \diagdown \quad \diagup B^\bullet \\ \quad \quad \quad \wp_i \\ \quad \quad \quad \\ \quad \quad \quad A \times_i B^\bullet \end{array}$	\times_i°	$\begin{array}{c} B^\circ \diagdown \quad \diagup A^\circ \\ \quad \quad \quad \times_i \\ \quad \quad \quad \\ \quad \quad \quad A \times_i B^\circ \end{array}$
\setminus_i^\bullet	$\begin{array}{c} A^\circ \diagdown \quad \diagup B^\bullet \\ \quad \quad \quad \times_i \\ \quad \quad \quad \\ \quad \quad \quad A \setminus_i B^\bullet \end{array}$	\setminus_i°	$\begin{array}{c} B^\circ \diagdown \quad \diagup A^\bullet \\ \quad \quad \quad \wp_i \\ \quad \quad \quad \\ \quad \quad \quad A \setminus_i B^\circ \end{array}$
$/_i^\bullet$	$\begin{array}{c} A^\bullet \diagdown \quad \diagup B^\circ \\ \quad \quad \quad \times_i \\ \quad \quad \quad \\ \quad \quad \quad A /_i B^\bullet \end{array}$	$/_i^\circ$	$\begin{array}{c} B^\bullet \diagdown \quad \diagup A^\circ \\ \quad \quad \quad \wp_i \\ \quad \quad \quad \\ \quad \quad \quad A /_i B^\circ \end{array}$

Figure 3.3: Links for Φ -LC

- Π is Φ -LC correct, i.e. it satisfies (DR_2) , $(ENDO)$, (BAL_i) for any modality i such that $a_i \notin \Phi$, and (ADJ_i) for any modality i such that $c_i \notin \Phi$; any switching of Π , restricted to its doors, is φ -equivalent to $\tau^\bullet \star A^\circ$;
- Π can be rewritten as $\tau^\bullet \star A^\circ$ using contractions, and the rewriting rules corresponding to the structural rules of the calculus;
- there is a Φ -LC derivation \mathcal{D} of $\tau \Rightarrow A$ such that $(\mathcal{D}) = \Pi$.

Moreover, we have seen that a (DR_2) -correct proof structure that is balanced for every modality satisfies necessarily the property $(ENDO)$.

Corollary 3.27 *Let Π be a Φ -LC proof structure and let $\tau \Rightarrow A$ be a Φ -LC sequent. The following facts are equivalent:*

- Π is Φ -LC correct, i.e. it satisfies (DR_2) , (BAL_i) and (ADJ_i) for any modality i ; any switching of Π , restricted to its doors, coincides with $\tau^\bullet \star A^\circ$;
- Π can be rewritten as $\tau^\bullet \star A^\circ$ using contractions;
- there is a Φ -LC derivation \mathcal{D} of $\tau \Rightarrow A$ that makes no use of structural rules and such that $(\mathcal{D}) = \Pi$.

Therefore, the correctness conditions for proof nets of the Calculus for headed trees ([71]) and, more in general, of any Φ -LC calculus with no structural rule are simply (DR_2) and, for any modality i , (BAL_i) and (ADJ_i) .

3.2 Bracketed Calculi

3.2.1 Linear Logic Φ -LLb

In this section, we present the logic Φ -LL of Section 3.1.1 enriched with a set of unary modalities that are not involved in any structural rule. Following G. Morrill's work on structural inhibition (see, e.g., [79]), the unary modalities are denoted by $[\]$ and $[\]^{-1}$, where $[\]^{-1}$ is the residual operator of $[\]$. Thus $[\]$ and $[\]^{-1}$ correspond to the modalities that are often denoted in the literature, respectively, as \diamond and \square^\perp (see, e.g., [54]).

Let \mathcal{N}_2 be a set of binary modalities and \mathcal{N}_1 be a set of unary modalities. Consider a subset Φ of $\{(ASS_i), (COM_i) : i \in \mathcal{N}_2\}$.

Definition 3.28 *The set $\mathcal{T}_{\Phi\text{-LLb}}(V)$ of Φ -LLb types is the smallest set that contains a set V of atomic types, their negations and that is closed, for any $i \in \mathcal{N}_2$, under the binary operations \times_i and \wp_i , and, for any $j \in \mathcal{N}_1$, under the unary operations $[j]$ and $[j]^{-1}$, i.e. the elements of $\mathcal{T}_{\Phi\text{-LL}}(V)$ are defined inductively in the following way (where $i \in \mathcal{N}_2$ and $j \in \mathcal{N}_1$):⁴*

$$\mathcal{T} = V \mid V^\perp \mid (\mathcal{T} \times_i \mathcal{T}) \mid (\mathcal{T} \wp_i \mathcal{T}) \mid [j]\mathcal{T} \mid [j]^{-1}\mathcal{T}.$$

A metalinguistic negation of types is defined in the following way. For every atomic type A , for any types F and G , and for any $i \in \mathcal{N}_2$ and $j \in \mathcal{N}_1$:

- $A = A^{\perp\perp}$;
- $(F \times_i G)^\perp = G^\perp \wp_i F^\perp$;
- $(F \wp_i G)^\perp = G^\perp \times_i F^\perp$;
- $([j]F)^\perp = [j]^{-1}F^\perp$;
- $([j]^{-1}F)^\perp = [j]F^\perp$.

As a consequence, $F = F^{\perp\perp}$ holds for any type F .

Definition 3.29 *A Φ -LLb sequent is an \mathcal{NL} seaweed where $\mathcal{N} = \mathcal{N}_2 \cup \mathcal{N}_1$ and $\mathcal{L} = \mathcal{T}_{\Phi\text{-LLb}}(V)$.*

⁴As usual, outer brackets will be dropped.

Definition 3.30 *The Φ -LLb calculus of sequents comprises the following rules, defined in Figure 3.4:*

- the identity and cut rules;
- the logical rules for binary operators;⁵
- the logical rules for unary operators;
- the structural rules that belong to Φ .

Identity and Cut rules			
Id	$\frac{}{A \star A^\perp}$ where A atomic	Cut	$\frac{\frac{\dot{D}_1}{\Gamma \star A} \quad \frac{\dot{D}_2}{A^\perp \star \Delta}}{\Gamma \star \Delta}$
Logical Rules of Binary Operators (for any $i \in \mathcal{N}_2$)			
Tensor	$\frac{\frac{\dot{D}_1}{\Gamma \star A} \quad \frac{\dot{D}_2}{B \star \Delta}}{[i \Gamma, A \times_i B, \Delta]}$	Par	$\frac{\dot{D}}{[i \Gamma, A, B]}$ $\frac{}{\Gamma \star A \wp_i B}$
Logical Rules of Unary Operators (for any $j \in \mathcal{N}_1$)			
Bracket	$\frac{\frac{\dot{D}}{\Gamma \star A}}{[j \Gamma, [j]A]}$	Antibracket	$\frac{\dot{D}}{[j \Gamma, A]}$ $\frac{}{\Gamma \star [j]^{-1}A}$
Structural rules			
(ASS_i)	$\frac{(\tau_0 \tilde{\times}_i \tau_1) \star (\tau_2 \tilde{\times}_i \tau_3)}{(\tau_1 \tilde{\times}_i \tau_2) \star (\tau_3 \tilde{\times}_i \tau_4)}$		
(COM_i)	$\frac{[i \tau_0, \tau_1, \tau_2]}{[i \tau_0, \tau_2, \tau_1]}$		

Figure 3.4: Rules of Φ -LLb

The usual proof of the Cut Elimination Property is easily extended to cover these logics.

3.2.2 Lambek Calculus Φ -LCb

In this section, we present Φ -LC enriched with a family of unary modalities, that are not involved in any structural rule. The embedding of Φ -LCb into

⁵Recall that for any binary modality i , the expression $[i \tau_0, \tau_1, \tau_2]$ denotes the seaweed $(\tau_0 \tilde{\times}_i \tau_1) \star \tau_2$, whereas for any unary modality j the expression $[j \tau_0, \tau_1]$ denotes the seaweed $[j] \tau_0 \star \tau_1$; see Definition 1.6.

Φ -LLb is given.

Let \mathcal{N}_2 be a set of binary modalities and \mathcal{N}_1 be a set of unary modalities. Consider a subset Φ of $\{(ASS_i), (COM_i) : i \in \mathcal{N}_2\}$.

Rules of Φ -LCb

Definition 3.31 *The set $\mathcal{T}_{\Phi\text{-LCb}}(V)$ of Φ -LCb types is the smallest set that contains a set V of atomic types and that is closed, for any $i \in \mathcal{N}_2$, under the binary operations \times_i , $/_i$, and \setminus_i , and for any $j \in \mathcal{N}_1$, under the unary operations $[j]$, $[j]^{-1}$ i.e. the elements of $\mathcal{T}_{\Phi\text{-LCb}}(V)$ are defined inductively in the following way (where $i \in \mathcal{N}_2$ and $j \in \mathcal{N}_1$):⁶*

$$\mathcal{T} = V \mid (\mathcal{T} \times_i \mathcal{T}) \mid (\mathcal{T} /_i \mathcal{T}) \mid (\mathcal{T} \setminus_i \mathcal{T}) \mid [j]\mathcal{T} \mid [j]^{-1}\mathcal{T}.$$

Let $\mathcal{N} = \mathcal{N}_2 \cup \mathcal{N}_1$ and $\mathcal{L} = \mathcal{T}_{\Phi\text{-LCb}}(V)$. Recall Definitions 1.1, 1.2 and 1.3 on page 19.

Definition 3.32 *An (abstract) Φ -LCb configuration is an (abstract) \mathcal{NL} -tree.*

Therefore abstract Φ -LCb configurations are defined inductively in the following way (where $i \in \mathcal{N}_2$ and $j \in \mathcal{N}_1$):

$$\tau = \mathcal{L} \mid (\tau \tilde{\times}_i \tau) \mid [\tilde{j}]\tau.$$

Definition 3.33 *An (abstract) Φ -LCb context is an (abstract) \mathcal{NL} -tree context.*

Therefore, abstract Φ -LCb contexts are defined inductively in the following way (where $i \in \mathcal{N}_2$, $j \in \mathcal{N}_1$ and τ is any Φ -LCb configuration):

$$\sigma[] = [] \mid \sigma[] \tilde{\times}_i \tau \mid \tau \tilde{\times}_i \sigma[] \mid [\tilde{j}]\sigma[].$$

Definition 3.34 *An (abstract) Φ -LCb sequent $\tau \Rightarrow A$ consists of an (abstract) Φ -LCb configuration τ and a Φ -LCb type A .*

Definition 3.35 (Φ -LCb calculus) *The calculus of Φ -LCb sequents comprises the following rules, defined in Figure 3.5:*

⁶As usual, outer parentheses (and) will be dropped.

- the identity and cut rules;
- the logical rules of binary and unary operators;
- the structural rules that belong to Φ .

Identity and cut rules			
Id	$\frac{}{A \Rightarrow A}$ where A atomic	Cut	$\frac{\frac{\dot{D}_1}{\tau \Rightarrow A} \quad \frac{\dot{D}_2}{\sigma[A] \Rightarrow B}}{\sigma[\tau] \Rightarrow B}$
Logical rules of binary operators (for any $i \in \mathcal{N}_2$)			
\times_i	$\frac{\frac{\dot{D}}{\sigma[A \tilde{\times}_i B] \Rightarrow C}}{\sigma[A \times_i B] \Rightarrow C}$	\times_i	$\frac{\frac{\dot{D}_1}{\tau_1 \Rightarrow A} \quad \frac{\dot{D}_2}{\tau_2 \Rightarrow B}}{\tau_1 \tilde{\times}_i \tau_2 \Rightarrow A \times_i B}$
\setminus_i	$\frac{\frac{\dot{D}_1}{\tau \Rightarrow A} \quad \frac{\dot{D}_2}{\sigma[B] \Rightarrow C}}{\sigma[\tau \tilde{\times}_i A \setminus_i B] \Rightarrow C}$	\setminus_i	$\frac{\frac{\dot{D}}{A \tilde{\times}_i \tau \Rightarrow B}}{\tau \Rightarrow A \setminus_i B}$
$/_i$	$\frac{\frac{\dot{D}_1}{\tau \Rightarrow A} \quad \frac{\dot{D}_2}{\sigma[B] \Rightarrow C}}{\sigma[B /_i A \times_i \tau] \Rightarrow C}$	$/_i$	$\frac{\frac{\dot{D}}{\tau \tilde{\times}_i A \Rightarrow B}}{\tau \Rightarrow B /_i A}$
Logical rules of unary operators (for any $j \in \mathcal{N}_1$)			
$[j]$	$\frac{\frac{\dot{D}}{\sigma[[j]A] \Rightarrow B}}{\sigma[[j]A] \Rightarrow B}$	$[j]$	$\frac{\frac{\dot{D}}{\tau \Rightarrow A}}{[j]\tau \Rightarrow [j]A}$
$[j]^{-1}$	$\frac{\frac{\dot{D}}{\sigma[A] \Rightarrow B}}{\sigma[[j][j]^{-1}A] \Rightarrow B}$	$[j]^{-1}$	$\frac{\frac{\dot{D}}{[j]\tau \Rightarrow A}}{\tau \Rightarrow [j]^{-1}A}$
Structural rules			
(ASS_i)	$\frac{\frac{\dot{D}}{\sigma[(\tau_1 \tilde{\times}_\varphi \tau_2) \tilde{\times}_\varphi \tau_3] \Rightarrow C}}{\sigma[\tau_1 \tilde{\times}_\varphi (\tau_2 \tilde{\times}_\varphi \tau_3)] \Rightarrow C}$		$\frac{\frac{\dot{D}}{\sigma[\tau_1 \tilde{\times}_\varphi (\tau_2 \tilde{\times}_\varphi \tau_3)] \Rightarrow C}}{\sigma[(\tau_1 \tilde{\times}_\varphi \tau_2) \tilde{\times}_\varphi \tau_3] \Rightarrow C}$
(COM_i)	$\frac{\frac{\dot{D}}{\sigma[\tau_0 \tilde{\times}_\varphi \tau_1] \Rightarrow C}}{\sigma[\tau_1 \tilde{\times}_\varphi \tau_0] \Rightarrow C}$		

Figure 3.5: Rules of the Φ -LCb calculus

Definition 3.36 We say that a Φ -LCb sequent $\tau \Rightarrow A$ is a theorem of

$\Phi\text{-LCb}$, and we write $\vdash_{\Phi\text{-LCb}} \tau \Rightarrow A$, if and only if an appropriate application of the rules of $\Phi\text{-LCb}$ yields a derivation

$$\frac{\mathcal{D}}{\tau \Rightarrow A}.$$

Embedding $\Phi\text{-LCb}$ into $\Phi\text{-LLb}$

Definition 3.37 A $\Phi\text{-ILLb}$ type, i.e. an intuitionistic $\Phi\text{-LLb}$ type, is defined by double induction based on a set V of atomic types (where $i \in \mathcal{N}_2$ and $j \in \mathcal{N}_1$):

$$\begin{aligned} \text{positive types: } & P = V \mid N_{\wp_i}P \mid P_{\wp_i}N \mid P \times_i P \mid [j]P \mid [j]^{-1}P; \\ \text{negative types: } & N = V^\perp \mid P \times_i N \mid N \times_i P \mid N_{\wp_i}N \mid [j]N \mid [j]^{-1}N. \end{aligned}$$

Definition 3.38 A $\Phi\text{-ILLb}$ sequent, i.e. an intuitionistic $\Phi\text{-LLb}$ sequent, is a $\Phi\text{-LLb}$ sequent, where one type is positive and the others are negative.

Definition 3.39 The output type A° (input type A^\bullet) is the translation of a $\Phi\text{-LCb}$ type A into a positive (negative) $\Phi\text{-LLb}$ type. A° is defined by induction as follows, whereas $B^\bullet = B^{\circ\perp}$ ($i \in \mathcal{N}_2, j \in \mathcal{N}_1$):

- $A^\circ = A$ if A is atomic;
- $A \times_i B^\circ = B^\circ \times_i A^\circ$;
- $A /_i B^\circ = B^\bullet \wp_i A^\circ$;
- $B \setminus_i A^\circ = A^\circ \wp_i B^\bullet$;
- $([j]A)^\circ = [j](A^\circ)$;
- $([j]^{-1}A)^\circ = [j]^{-1}(A^\circ)$.

Since a double negation cancels out, for any type B one has that $B^\circ = B^{\bullet\perp}$ and therefore:

- $A^\bullet = A^\perp$ if A is atomic;
- $A \times_i B^\bullet = A^\bullet \wp_i B^\bullet$;
- $A /_i B^\bullet = A^\bullet \times_i B^\circ$;

- $B \setminus_i A^\bullet = B^\circ \times_i A^\bullet$;
- $([j]A)^\bullet = [j]^{-1}(A^\bullet)$;
- $([j]^{-1}A)^\bullet = [j](A^\bullet)$.

The translation extends to Φ -**LCb** configurations.

Definition 3.40 *The intuitionistic translation of a Φ -**LCb** sequent $\tau \Rightarrow A$ is the intuitionistic Φ -**LLb** sequent $\tau^\bullet \star A^\circ$, where τ^\bullet is defined as follows:*

- if $\tau = B$, then $\tau^\bullet = B^\bullet$;
- if $\tau = \tau_0 \tilde{\times}_i \tau_1$, then $\tau^\bullet = \tau_0^\bullet \tilde{\times}_i \tau_1^\bullet$;
- if $\tau = \tilde{[j]}\tau_1$, then $\tau^\bullet = \tilde{[j]}(\tau_1^\bullet)$.

The same reasoning expounded for previous analogous results establishes the following result.

Theorem 3.41 *A Φ -**LCb** sequent $\tau \Rightarrow A$ is a theorem of Φ -**LCb** if and only if its intuitionistic translation $\tau^\bullet \star A^\circ$ is a theorem of Φ -**LLb**.*

The translation of derivations preserves the use of the cut rule (or lack thereof).

Corollary 3.42 *The calculus Φ -**LCb** enjoys the Cut Elimination Property.*

3.2.3 Mimicking brackets with two extra symbols

In this section, we propose a theory of proof nets for **LCb** based on the use of two extrasymbols that are proxies for open and close bracket.

Consider the set \mathcal{T}_b of **LCb**-types defined on the basis of a set \mathcal{V} of atomic types and a singleton set of unary modalities. Let \mathcal{T} be the set of **LC**-types defined on the basis of $\mathcal{V} \cup \{P, Q\}$, where P and Q are auxiliary symbols that do not belong to \mathcal{V} .

Definition 3.43 *Let t be the function from \mathcal{T}_b to \mathcal{T} defined in the following way:*

- $t(A) = A$ if A is atomic;
- $t(A \times B) = t(A) \times t(B)$;
- $t(A/B) = t(A)/t(B)$;
- $t(A \setminus B) = t(A) \setminus t(B)$;
- $t(\llbracket A \rrbracket) = P \times t(A) \times Q$;
- $t(\llbracket^{\perp} A \rrbracket) = P \setminus t(A) / Q$.

The translation function t can be extended to sequents:

- $t(\Gamma, \Delta) = t(\Gamma), t(\Delta)$;
- $t(\llbracket \Gamma \rrbracket) = P, t(\Gamma), Q$.

Under this translation, bracket and antibracket rules are compilations of product- and slash-rules. Thus the translation t preserves theoremhood. Moreover, the translation of an **LCb**-proof results in a **LC**-proof where the auxiliary symbols P and Q are *matched up properly*, i.e. the axioms introducing the symbols P and Q come in pairs so that the two symbols arising from the translation of a unary operator are matched up with the two symbols arising from the translation of one and a single unary operator.

The translation t does not in general reflect theoremhood, for there are non-theorems⁷ of **LCb**, such as $\llbracket \llbracket^{\perp} A \rrbracket, \llbracket \llbracket^{\perp} B \rrbracket \rrbracket \Rightarrow \llbracket \llbracket \llbracket^{\perp} (A \times B) \rrbracket \rrbracket$, that are translated into theorems of **LC**:

$$\begin{array}{c}
\frac{\frac{\frac{\frac{\overline{P \Rightarrow P}}{P \Rightarrow P} \quad A, B \Rightarrow A \times B}{P, P \setminus A, B \Rightarrow A \times B}}{P \setminus A, B \Rightarrow P \setminus (A \times B)}}{P \setminus A, P, P \setminus B \Rightarrow P \setminus (A \times B)} \quad \overline{Q \Rightarrow Q}}{\frac{P \setminus A, P, (P \setminus B) / Q, Q \Rightarrow P \setminus (A \times B)}{P \setminus A, P, (P \setminus B) / Q \Rightarrow (P \setminus (A \times B)) / Q} \quad \overline{Q \Rightarrow Q}}{\frac{P \setminus A, P, (P \setminus B) / Q, Q \Rightarrow ((P \setminus (A \times B)) / Q) \times Q}{P, (P \setminus A) / Q, Q, P, (P \setminus B) / Q, Q \Rightarrow P \times (((P \setminus (A \times B)) / Q) \times Q)} \quad \overline{Q \Rightarrow Q}}
\end{array}$$

⁷Observe that, simulating a necessity operator $\Box A$ as $\llbracket \llbracket^{\perp} A \rrbracket$ (see [69]), the derivation presented below leads to a proof of the translation of the K axiom $\Box A \times \Box B \Rightarrow \Box(A \times B)$. However, this translation is not useful in dealing with K-like logics, since a similar derivation yields a proof of the translation of $\Box A \times (C \times \Box B) \Rightarrow \Box(A \times (C \times B))$.

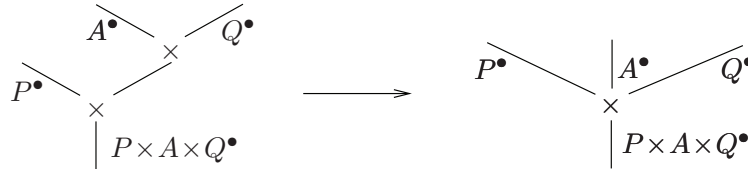
The previous example shows, contra [122], that this property is not enough to guarantee that the translation t reflects theoremhood. However, as proved in [122], in any **LC**-proof the inference steps leading from a type A to a type $P \times A \times Q$ or $P \setminus A / Q$ can be grouped together. The gist of the proof is that if the inference rule introducing, say, $P \setminus A$ (or $P \times A$) occurs earlier in a proof, then the rule introducing $P \setminus A / Q$ (respectively, $P \times A \times Q$) can be pushed up so as to become the immediately successive rule. The proof is by induction on the length of a proof of the sequent. The crucial observation is the permutability of rules as in the following example, where the left rule of the slash operator is permuted up past the right product rule:

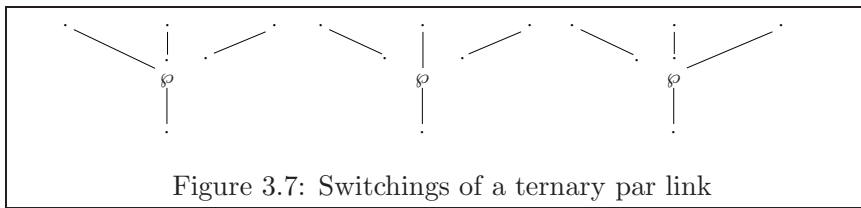
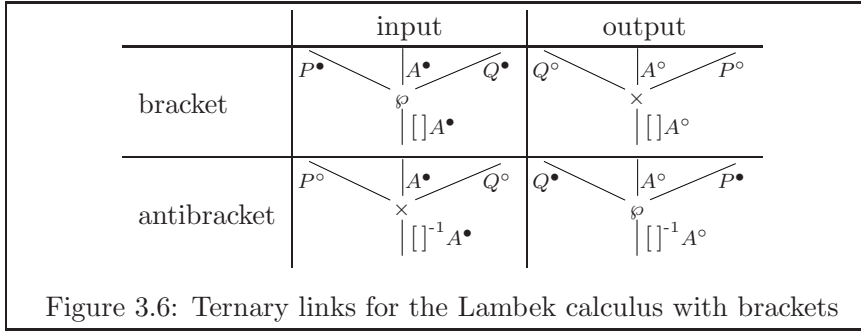
$$\frac{\frac{\frac{\vdots}{\Gamma \Rightarrow B} \quad \frac{\vdots}{\Delta(P \setminus A) \Rightarrow C}}{\Gamma, \Delta(P \setminus A) \Rightarrow B \times C}}{\frac{Q \Rightarrow Q}{\Gamma, \Delta(P \setminus A / Q, Q) \Rightarrow B \times C}} \quad \frac{\frac{\frac{\vdots}{Q \Rightarrow Q} \quad \frac{\vdots}{\Delta(P \setminus A) \Rightarrow C}}{\Gamma, \Delta(P \setminus A / Q, Q) \Rightarrow C}}{\Gamma, \Delta(P \setminus A / Q, Q) \Rightarrow B \times C}}$$

Proposition 3.44 *A sequent $\vdash \Gamma \Rightarrow A$ of **LCb** is a theorem if and only if its translation $\vdash t(\Gamma) \Rightarrow t(A)$ has a proof in **LC** where the axioms introducing the types P and Q come in pairs so that the types P and Q arising from the translation of one unary operator are matched up with the two types P and Q arising from the translation of one and a single unary operator.*

Proof. The necessity of the condition can be verified by a simple proof by induction on the length of **LCb**-proofs. The condition is sufficient because the inference rules leading from a type A to a type $P \times A \times Q$ or $P \setminus A / Q$ can be grouped together and because splittings preserve the proper matching of the auxiliary symbols P and Q . \square

The previous proposition yields a theory of proof nets for **LCb**. An **LC**-proof structure corresponds to the translation under t of an **LCb**-proof if and only if it is a proof net of **LC** in which moreover the symbols P and Q are properly matched up. To an (anti)bracketed type is associated a tree composed by two binary links of the same type that can therefore be compiled away, as in the following example, in a single ternary link of the same type:





Some modification of basic definitions are in order:

- (partial) proof structures/pseudostructures can have 4-ary nodes: they are labeled by elements of $\mathcal{N}_{[]} \cup \mathcal{N}_{[]^{-1}}$; incident edges are ordered cyclically; one of them is specified to be the conclusion; the only premise labeled by a type of **LCb** is called the proper premise;
- beside the usual links of the Lambek Calculus, the repertoire of links is enriched with four ternary links for brackets and antibrackets, see Figure 3.6;
- a switching of a \emptyset -link, just as for the Lambek Calculus, is simply the choice of one of the premises of the link; but the notion of switched proof structure has to be changed slightly, taking into account the fact that, in any switching of a ternary link, there is more than one premise that is not selected. The new definition is: a switching $s\Pi$ of a **LCb** structure Π is the graph that results from a switching of its \emptyset -links by disconnecting from each \emptyset -node the *premises* that *are* not selected by the switching, as indicated in Figure 3.7.

Note that the compilation of the two binary links into a ternary link does not affect the connectedness and acyclicity of switchings. These observations establish the correctness criteria for **LCb**-proof nets and, by the same token, show that two **LCb**-derivations are \approx -equivalent if and only if they are mapped to the same **LCb**-proof net.

Theorem 3.45 *Let Π be an **LCb** proof structure and $\tau \Rightarrow A$, an **LCb** sequent. The following facts are equivalent:*

- *There is an **LCb** derivation \mathcal{D} of $\tau \Rightarrow A$ such that $(\mathcal{D}) = \Pi$;*
- *Π satisfies the following correctness criteria:*
 1. *every switching of Π is acyclic;*
 2. *Π has exactly one output door;*
 3. *every trimming of a par link contains a door of Π ;*
 4. *Π satisfies condition (PL) ;*
 5. *brackets are matched up correctly.*

The condition on uniqueness of an output door can be substituted by the requirement that every switching of Π is connected.

The condition (PL) can be substituted by the adjacency condition, provided that we adapt the relevant terminology to the ternary links. Observe that:

- the notion of trimming of a par link needs no modification, since it is about the disconnection of the selected premise of the link;
- for any (DR) -correct proof structure, the premises of a ternary link L are joined pairwise by a path in any trimming at L .

Definition 3.46 *Let Π be an **LCb** structure that satisfies (DR_2) . Let τ be a trimming of Π at a ternary par link L . Consider a door D of Π that belongs to τ . The paths that join pairwise D and the two left-most premises of L meet at $l\text{-center}_\tau(L, D)$, a node called the left center in τ of L and D . The paths that join pairwise D and the two right-most premises of L meet at $r\text{-center}_\tau(L, D)$, a node called the right center in τ of L and D .*

Definition 3.47 *We say that a door D is not between the premises of L in τ if:*

- *the paths that join $l\text{-center}_\tau(L, D)$ with, respectively, the first and second premise of L and D are in this cyclic order when moving counterclockwise around $l\text{-center}_\tau(L, D)$;*
- *the paths that join $r\text{-center}_\tau(L, D)$ with, respectively, the second and third premise of L and D are in this cyclic order when moving counterclockwise around $r\text{-center}_\tau(L, D)$.*

are based on an embedding into an endomodal Φ -**LL** logic enriched with one commutative modality. The theory transfers to Φ -**LC** calculi enriched with a unary modality, via the usual embedding into the intuitionistic fragment of the logic.

Let \mathcal{L} be a Φ -**LLb** logic based on a set V of atomic symbols and that has one unary modality j and a set I of binary modalities. Let V' be $V \cup \{P\}$, where $P \notin V$, and let I' be the set I of binary modalities enriched with a further modality \times_j that is commutative and associative. Consider the Φ -**LL** logic \mathcal{L}' defined on the set V' of atomic symbols and comprising the set I' of binary modalities, where every modality of I has the same properties that it had in \mathcal{L} .

The translation s of \mathcal{L} types as \mathcal{L}' types is defined as follows:

- $sA = A$ if A is atomic or the negation of an atomic type;
- $s(A \circ_i B) = sA \diamond_i sB$ where $\circ \in \{\times, \wp\}$ and $i \in I$;
- $s([j]A) = P \times_j sA$;
- $s([j]^{-1}A) = P^\perp \wp_j sA$.

A simple proof by induction over the complexity of \mathcal{L} -types shows that $s(A^\perp) = (sA)^\perp$ for any \mathcal{L} -type A .


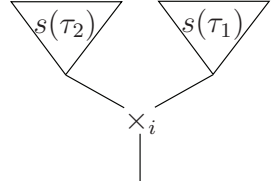
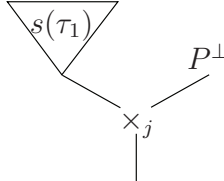
The translation s extends to trees setting:

$$s\tau = \begin{cases} sA & \text{if } \tau \text{ is a type } A; \\ s\tau_1 \tilde{\times}_i s\tau_2 & \text{if } \tau = \tau_1 \tilde{\times}_i \tau_2; \\ P^\perp \tilde{\times}_j s(\tau_1) & \text{if } \tau = [j]\tau_1. \end{cases}$$

Observe that for any associative modality h and for any commutative modality k the following equations hold by definition of \mathcal{L}' (where τ_1 , τ_2 , and τ_3 are any \mathcal{L} -tree):

$$\begin{aligned} s\tau_1 \tilde{\times}_h (s\tau_2 \tilde{\times}_h s\tau_3) &= (s\tau_1 \tilde{\times}_h s\tau_2) \tilde{\times}_h s\tau_3 \\ s\tau_1 \tilde{\times}_k s\tau_2 &= s\tau_2 \tilde{\times}_k s\tau_1 \end{aligned}$$

Graphically, we have:

$s(A)$	$s(\tau_1 \tilde{\times}_i \tau_2)$	$s([\tilde{j}] \tau_1)$
$s(A)$ 		

Observe that in any translation $s\alpha$ of an \mathcal{L} -tree α there are as many $\tilde{\times}_j$ -nodes as leaves labeled by P^\perp . If $s\alpha = \tau_1 \tilde{\times}_i \tau_2$, then the above mentioned property holds for each of the subtrees τ_1 and τ_2 , because there is no mixed-associative law involving the modalities i and j .

Note that for any tree τ_j ($j \in 1, 2, 3$) and any binary modality $i \in I$, the following equalities hold:

$$s(\tau_1 \tilde{\times}_i \tau_2) \star s\tau_3 = s\tau_1 \star s(\tau_2 \tilde{\times}_i \tau_3) = s\tau_2 \star s(\tau_3 \tilde{\times}_i \tau_1).$$

$$s[\tilde{j} \tau_1] \star s\tau_2 = s\tau_1 \star s[\tilde{j} \tau_2]$$

Indeed, the first equation holds because the translation s is a homomorphism with respect to binary structural operations and therefore each of its terms is equal to $[\tilde{j} s\tau_1, s\tau_2, s\tau_3]$. The second equation holds because the j -modality is commutative and therefore $[\tilde{j} P^\perp, s\tau_1, s\tau_2] = [\tilde{j} s\tau_1, P^\perp, s\tau_2]$. Observe also that associativity, or lack thereof, of the j -modality is immaterial for the argument.

Thus the translation s can be extended to \mathcal{L} sequents (where A and B are types; τ_1 , τ_2 , and τ_3 are trees; and $i \in I$):

$$\bullet \quad s\alpha = \begin{cases} sA \star sB & \text{if } \alpha = A \star B; \\ [\tilde{i} s\tau_1, s\tau_2, s\tau_3] & \text{if } \alpha = [\tilde{i} \tau_1, \tau_2, \tau_3]; \\ [\tilde{j} P^\perp, s\tau_1, s\tau_2] & \text{if } \alpha = [\tilde{j} \tau_1, \tau_2]. \end{cases}$$

Graphically, we have:

$s(A \star B)$	$s[_i \tau_1, \tau_2, \tau_3]$	$s[_j \tau_1, \tau_2]$
sA \downarrow sB		

Thus any translation $s\alpha$ of an \mathcal{L} -seaweed α satisfies the following properties (where A_1 and A_2 are \mathcal{L} -types and τ_1 , τ_2 , and τ_3 are \mathcal{L} -trees):

- if $s\alpha$ is the trivial seaweed $A_1 \star A_2$ then, for any $h \in \{1, 2\}$, $A_h = sB_h$ for some suitable \mathcal{L} -type B_h ; more generally, all the leaves of a translation $s\alpha$ are translations of \mathcal{L} -types;
- if $s\alpha = [_i \tau_1, \tau_2, \tau_3]$ then, for any $h \in \{1, 2, 3\}$, τ_h is equal, up to associativity and commutativity of the modality j , to $s\sigma_h$ for some suitable \mathcal{L} -tree σ_h ;
- if $s\alpha = [_j P^\perp, \tau_1, \tau_2]$ then, for any $h \in \{1, 2\}$, τ_h is equal, up to associativity and commutativity of the modality j , to $s\sigma_h$ for some suitable \mathcal{L} -tree σ_h .

Observe that the **LCb** non-theorem mentioned earlier, i.e.

$$[[\]^{-1}A], [[\]^{-1}B] \Rightarrow [[\]^{-1}(A \times B)$$

yields, under the translation, a sequent that clearly cannot be proved in the new setting:

$$(P \tilde{\times}_j (P^\perp \wp_j A)) \tilde{\times} (P \tilde{\times}_j (P^\perp \wp_j B)) \Rightarrow P \times_j (P^\perp \wp_j (A \times B))$$

Theorem 3.49 *Let α be an \mathcal{L} sequent. Then α is an \mathcal{L} theorem if and only if its translation $s\alpha$ is an \mathcal{L}' theorem.*

Proof. The necessity of the condition can be verified by a straightforward proof by induction on the length of \mathcal{L} -derivations. To show that the condition is sufficient, we will proceed by induction over the number n of logical

rules in cut-free \mathcal{L}' -derivations. If n is zero, there is nothing to show. Suppose that a derivation \mathcal{D} of $s\alpha$ involves at least one application of a logical rule and consider the bottom-most logical rule r . If r is an instance of a binary rule of modality $i \in I$, then the derivation has the following structure, where A_0 and A_1 are \mathcal{L} -types, τ_0 and τ_1 are \mathcal{L}' -trees, and Σ is a sequence of structural rules:

$$\frac{\frac{\frac{\vdots D'_0}{\tau_0 \star sA_0}}{\frac{\vdots D'_1}{sA_1 \star \tau_1}}}{[i \tau_0, sA_0 \times_i sA_1, \tau_1]} \frac{\vdots \Sigma}{s\alpha}$$

There are \mathcal{L} -trees σ_0 and σ_1 such that, for any $h \in \{0, 1\}$, the translated tree $s\sigma_h$ is equal, up to \mathcal{L}' -structural rules, to τ_h and such that $s\alpha = [i \ s\sigma_0, s(A_0 \times_i A_1), s\sigma_1]$. For any $h \in \{0, 1\}$, continuing the derivation D'_h with suitable structural rules yields a \mathcal{L}' -derivation of $s\sigma_h \star sA_h$ and therefore, by induction hypothesis, there are derivations D_h of $\sigma_h \star A_h$. Combining D_1 and D_2 with the \times_i -rule yields finally a derivation of α . If the last logical rule r is an instance of a \times_j -rule, then the structure of the derivation is the following, where A is an \mathcal{L} -type, τ_0 and τ_1 are \mathcal{L}' -trees, and Σ is a list of structural rules:

$$\frac{\frac{\frac{\vdots D'_0}{\tau_0 \star P}}{\frac{\vdots D'_1}{sA \star \tau_1}}}{[j \ \tau_0, P \times_j sA, \tau_1]} \frac{\vdots \Sigma}{s\alpha}$$

Observe that the derivation D'_0 can only be trivial, meaning that τ_0 must be P^\perp . Indeed, the P^\perp that is the other conclusion of the axiom that introduces P has to be combined in D'_0 by a tensor rule, the only way to join subderivations in a cut-free proof. But this would yield a type $B \times_h P^\perp$, that cannot be a subtype of any translated \mathcal{L} -type. Therefore $\tau_0 = P^\perp$ and there is an \mathcal{L} -tree σ_1 such that $s\sigma$ is equal, up to \mathcal{L}' -structural rules, to τ_1 and such that $s\alpha = [j \ [j]A, \sigma_1]$. Completing D'_1 at the bottom with suitable structural rules yields a derivation of $sA \star s\sigma_1$. By induction hypothesis there is an \mathcal{L} -derivation D_1 of $A \star \sigma_1$ and therefore, simply applying at the end a $[j]$ -rule, an \mathcal{L} -derivation of $[j \ [j]A, \sigma_1]$. If the last logical rule r is unary the derivation has one of the following structures, where A_0 and A_1 are \mathcal{L} -types,

τ is an \mathcal{L}' -tree and Σ is a list of \mathcal{L}' -structural rules:

$$\frac{\frac{\frac{\vdots \mathcal{D}'}{[{}_i \tau, sA_0, sA_1]}{\tau \star (sA_0) \wp_i (sA_1)}}{\vdots \Sigma}}{s\alpha}}{\frac{\frac{\frac{\vdots \mathcal{D}'}{[{}_j \tau, P^{\perp}, sA_1]}{\tau \star P^{\perp} \wp_i (sA_1)}}{\vdots \Sigma}}{s\alpha}}$$

In any case there is an \mathcal{L} -tree σ such that $s\sigma$ is equal, up to \mathcal{L}' -structural rules, to τ . Moreover, in the first case $\alpha = \sigma \star A_0 \wp_i A_1$ and in the second case $\alpha = \sigma \star [j]^{-1} A_1$. Applying the list Σ of structural rules immediately after the derivation D' yields an \mathcal{L}' -derivation of, respectively, $s[{}_i \sigma, A_0, A_1]$ and $s[{}_j \sigma, A_1]$. In each case, by induction hypothesis, there is a derivation D of $[{}_i \sigma, A_0, A_1]$ (respectively $[{}_j \sigma, A_1]$) that yields, adding a \wp_i -rule ($[j]^{-1}$ -rule), an \mathcal{L} -derivation of α . \square

The previous result on the faithfulness of the embedding and the theory already established for Φ -LL calculi, yield the following results without the need for any other proof.

Definition 3.50 *A Φ -LLb pseudostructure/(partial) proof structure is an $\mathcal{N}\mathcal{L}$ pseudostructure/(partial) proof structure where $\mathcal{N} = I \cup \{j\}$, and $\mathcal{L} = \mathcal{T}_{\Phi\text{-LLb}}(V)$.*

Corollary 3.51 *Let Π be a Φ -LLb proof structure and let α be a Φ -LLb sequent. The following facts are equivalent:*

- Π is Φ -LLb correct, i.e. it satisfies (DR_2) , $(ENDO)$, (BAL_i) for any binary modality i such that $a_i \notin \Phi$, and (ADJ_i) for any binary modality i such that $c_i \notin \Phi$; any switching of Π is Φ -LLb-equivalent to α ;
- there is a Φ -LLb derivation \mathcal{D} of α such that $(\mathcal{D}) = \Pi$.

The embedding of Φ -LCb into Φ -LLb allows us to transfer the previous characterization to the former calculi.

Corollary 3.52 *Let Π be a Φ -LCb proof structure and let $\tau \Rightarrow A$ be a Φ -LCb sequent. The following facts are equivalent:*

- Π is Φ -**LCb** correct, i.e. it satisfies (DR_2) , $(ENDO)$, (BAL_i) for any binary modality i such that $a_i \notin \Phi$, and (ADJ_i) for any binary modality i such that $c_i \notin \Phi$; any switching of Π is Φ -**LLb**-equivalent to $\tau^\bullet \star A^\circ$;
- there is a Φ -**LCb** derivation \mathcal{D} of $\tau \Rightarrow A$ such that $(\mathcal{D}) = \Pi$.

Part II

Beyond multimodality

Chapter 4

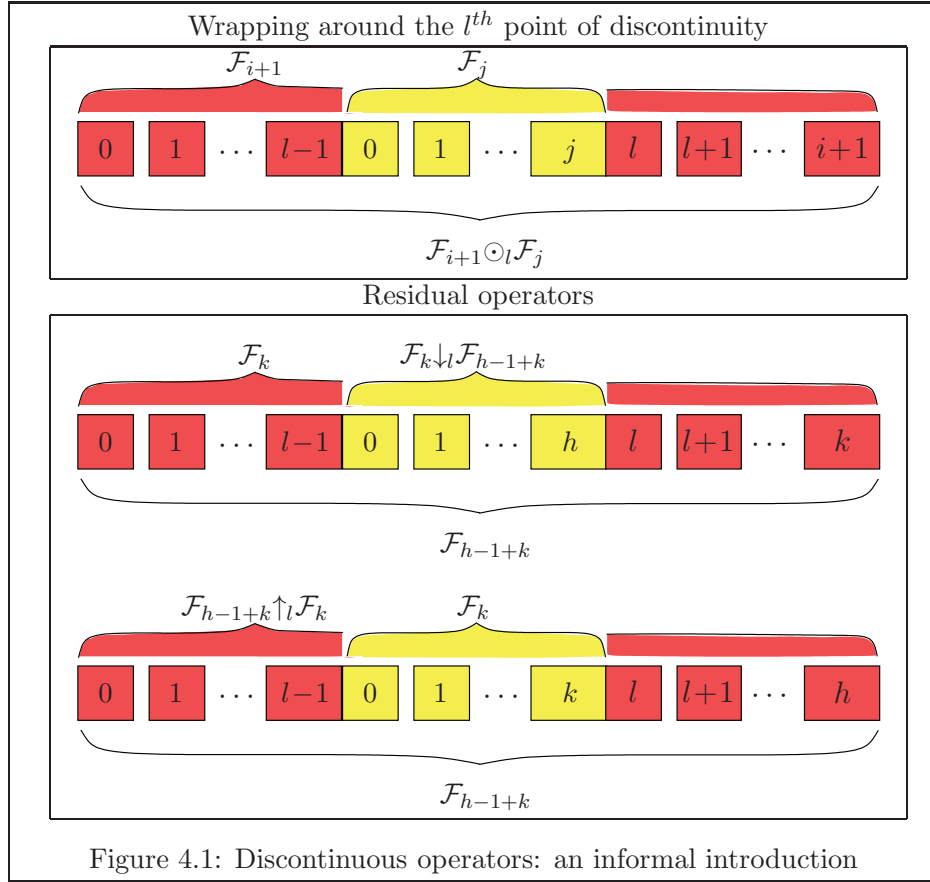
Discontinuous Lambek Calculus

The Lambek Calculus **LC** is a deductive system endowed with a multiplicative conjunction, called henceforth the continuous product, and its residual operators, the backward and forward slashes. It is a logic for typing trees identified up to associativity and deals, therefore, with lists of types. On the phonological level, the continuous product is interpreted in free semigroups as the operation of concatenation. But natural language goes beyond concatenation and includes discontinuous phenomena, i.e. it displays semantic units that are realized phonetically by non-adjacent material and, more generally, semantic-prosodic mismatches.

The research on the Discontinuous Lambek Calculus ([82, and references therein]) aims at formulating a logic that keeps as many as possible interesting properties of **LC**, while being able to deal in a natural way with discontinuity. Applications of variants of the Discontinuous Lambek Calculus have been discussed in a number of publications ([79, 84, 91, 90, 82, 87, 88]), some of which written in collaboration with the author.

The core of the Discontinuous Lambek Calculus is the Hypersequent Calculus $\mathbf{HC}_\epsilon^\omega$, a logic built around a deterministic discontinuous product called *wrap*.¹ In these systems, a type with i points of discontinuity is encoded in a configuration by the list of its $i + 1$ components. The discontinuous

¹A discontinuous type constructor was proposed in [68], but the sequent calculus lacked a left rule for the introduction of the operator. An ordered sequent format for wrap was given in [78] and a prosodically labelled sequent format in [79]. Earlier, Pollard's thesis ([102]) had proposed wrapping of *headed* strings, the head of the second string being the target of the wrapping (see [119] for a discussion of the weak equivalence of Head Grammars and grammars based on a fragment of the Discontinuous Lambek Calculus).



products combine types via wrapping avoiding interleaving of components of different types, as sketched informally in Figure 4.1. Concatenation can be seen as a degenerate case of wrapping, namely when all the components of the first factor precede the components of the second. The *extract* and *infix* implications (see Figure 4.1) are the residual operations of the wrap product.²

Using the vector notation, proposed by the author in the joint work [84] (see Section 4.2.1), the logical rules of the calculus follow the same schema as the **LC** rules (see Section 4.2.2). As for the interaction with semantics, there is nothing to add, since the discontinuous product and implications are treated just as the Lambek product and slashes. The Cut Elimination Property has been proved for $\mathbf{HC}_\epsilon^\omega$ ([90]) and for the *Displacement Calculus*, i.e. $\mathbf{HC}_\epsilon^\omega$

²The extract and infix implications appear already in [68] following the work of [4, 5]. For a sequent presentation of these operators –with control over the insertion point– see again [78, 79].

enriched with units of sort zero and one.

Another version of the Hypersequent Calculus can be obtained enriching $\mathbf{HC}_\epsilon^\omega$ with a nondeterministic wrap operator and its two residual operators ([84]). The nondeterministic wrap, represented by the operator \odot , has an additive flavour. To introduce on the left a type $A\odot B$, one must be able to deterministically wrap A around B at any point of discontinuity.³

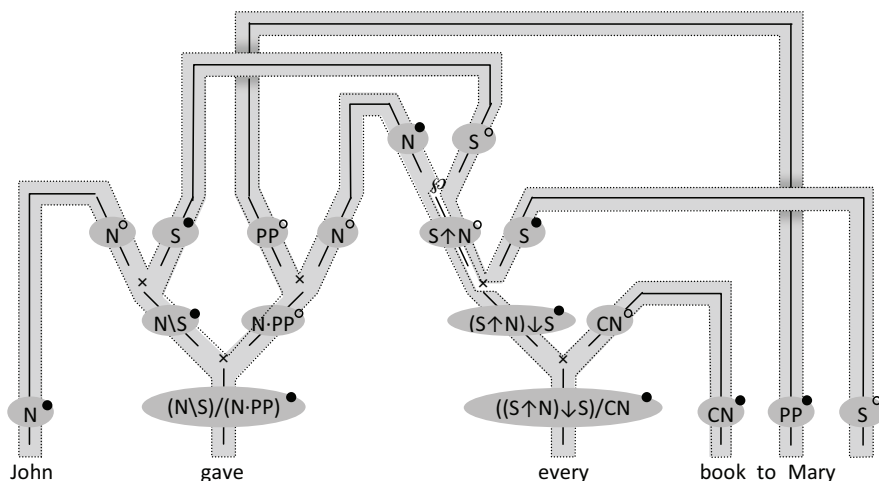
Phonologically, discontinuous types are interpreted in free graded algebras, i.e. free monoids built out of a collection of prime elements, one of which –the separator– stands for the point of discontinuity. Completeness with respect to these models is an open question. For the fragment of the calculus with at most one point of discontinuity, [119] shows completeness with respect to powersets of 1-graded residuated algebras. The only known result about the generative power of Hypersequent Calculi, is a lower bound for the Displacement Calculus mentioned earlier: this calculus recognizes the permutation closures of context free languages ([89]).

In this chapter, we present an unpublished theory of proof nets for $\mathbf{HC}_\epsilon^\omega$ and its variants $\mathbf{HC}_{\dagger_0}^\omega$ and $\mathbf{HC}_{\ddagger}^\omega$, which are obtained from the former barring the empty configuration with no point of discontinuity and, respectively, all the empty configurations which consist solely of $n \geq 0$ points of discontinuity.

The basic idea is that proof structures for Hypersequent Calculi are essentially proof nets of \mathbf{LCP} , the Lambek Calculus with Permutation, enriched with *trip instructions*. In the basic case, when we consider types that have at most one point of discontinuity, we can think that links in such proof structures come equipped with ribbons, the edges of which are the trip instructions. The general case is conceptually similar, although it is more difficult to represent. In any case, the trip instructions determine, in any proof net, a *prosodic trip* i.e. a way of travelling deterministically through its conclusions. Moreover, the proof net can be sequentialized as a derivation of a sequent in which the order of (components of) types corresponds to the order of visiting them along the prosodic trip.

We illustrate these ideas with the following example. In this case the prosodic-semantic mismatch arises from a quantifier phrase that takes sentential scope semantically, even if in a traditional grammar it would be considered a complement of the verb. Observe that the prosodic trip (the dotted cycle) travels through the conclusions in the given linear order of the conclusions, despite the fact that the proof structure is not planar.

³As pointed out by one of the reviewers, $\mathbf{HC}_\epsilon^\omega$ seems to be to \mathbf{LC} what Multiple Context Free Grammars are to Context Free Grammars. Addition of the nondeterministic wrap disrupt the simile.



The chapter is structured as follows. To facilitate the comprehension of the general theory, Section 4.1 illustrates –somewhat informally– the main ideas on the fragment **BDLC** of the calculus $\mathbf{HC}_\varepsilon^\omega$, where types have at most one point of discontinuity and *parameter paths* can be pictured easily. This serves also the purpose of presenting a corrected version of the theory expounded in the joint work with G. Morrill ([83]).

The approach followed in [83] consisted of two steps. Firstly, we extended to **BDLC** Moot and Piazza’s embedding of **LC** into First Order Intuitionistic Linear Logic ([76]), thereby bringing into syntax a relational interpretation à la van Benthem ([121]). Secondly, we proposed a characterization of those proof structures that represent translations of **BDLC** derivations.

Although it is, in principle, possible to extend this approach to cover the case of an unbounded number of discontinuities, the sheer number of parameters involved in the translation dissuades from trying. However, it is clear now that reference to quantification can be obviated altogether, following an approach that is akin to Melliès’s topological characterization ([67]) of proof nets for Non-Commutative Multiplicative Linear Logic ([2, 112]). Essentially our *parameter paths* correspond, in his theory, to the borders of the *ribbons*. But instead of modifying the parameter paths when switching a proof structure, we use them to define a generalized notion of switching. The key concept is the notion of hyperswitching, inspired by Girard’s notion of *jump* ([41]) and Bellin and van Wiele’s subsequent work ([10]). Indeed Section 4.2 can be seen as an adaptation of the latter publication to the discontinuous calculus.

4.1 The Hypersequent Calculus BDLC

In this section we present briefly the theory for the Hypersequent Calculus **BDLC**, the language of which is obtained enriching the **LC** language with types of sort 1, and with a wrap operator that yields continuous types and its residual implications.

Definition 4.1 *The types \mathcal{F}_0 of sort 0 and \mathcal{F}_1 of sort 1 are defined on the basis of a set \mathcal{A} of atomic types of sort 0 by:*

$$\begin{aligned}\mathcal{F}_0 &::= \mathcal{A} \mid \mathcal{F}_0 \cdot \mathcal{F}_0 \mid \mathcal{F}_0 \setminus \mathcal{F}_0 \mid \mathcal{F}_0 / \mathcal{F}_0 \mid \mathcal{F}_1 \odot \mathcal{F}_0 \mid \mathcal{F}_1 \downarrow \mathcal{F}_0 \\ \mathcal{F}_1 &::= \mathcal{F}_0 \uparrow \mathcal{F}_0\end{aligned}$$

The connectives \cdot , \setminus and $/$ are called ‘continuous product’, ‘under’ and ‘over’, and the connectives \odot , \downarrow and \uparrow are called ‘discontinuous product’, ‘infix’ and ‘extract’. A type \mathcal{F}_1 of sort 1 has two components $\sqrt[0]{\mathcal{F}_1}$ and $\sqrt[1]{\mathcal{F}_1}$.

Definition 4.2 *The configurations \mathcal{O}_0 of sort 0 and \mathcal{O}_1 of sort 1 are defined by the following unambiguous grammar, where Λ is the empty configuration and $[]$ is the metalogical separator marking the point of discontinuity:*

$$\begin{aligned}\mathcal{O}_0 &::= \Lambda \mid \mathcal{F}_0, \mathcal{O}_0 \mid \sqrt[0]{\mathcal{F}_1}, \mathcal{O}_0, \sqrt[1]{\mathcal{F}_1}, \mathcal{O}_0 \\ \mathcal{O}_1 &::= \mathcal{O}_0, [], \mathcal{O}_0 \mid \mathcal{O}_0, \sqrt[0]{\mathcal{F}_1}, \mathcal{O}_1, \sqrt[1]{\mathcal{F}_1}, \mathcal{O}_0\end{aligned}$$

Definition 4.3 *A **BDLC** sequent $\mathcal{O}_i \Rightarrow \mathcal{F}_i$ of sort i ($i \in \{0, 1\}$) consists of a **BDLC** configuration \mathcal{O}_i and a **BDLC** type \mathcal{F}_i , both of sort i .*

For any type \mathcal{F}_1 of sort 1, the vector notation $\vec{\mathcal{F}}_1$ designates the configuration $\sqrt[0]{\mathcal{F}_1}, [], \sqrt[1]{\mathcal{F}_1}$. For any configuration $\mathcal{O}_1 = \Gamma, [], \Delta$ of sort 1 and \mathcal{O}_0 of sort 0, the notation $\mathcal{O}_1 \mid \mathcal{O}_0$ denotes the configuration $\Gamma, \mathcal{O}_0, \Delta$.

The expression $\Gamma[\mathcal{F}_0]$ indicates a configuration with a distinguished occurrence of the type \mathcal{F}_0 of sort zero. $\Gamma[\vec{\mathcal{F}}_1]$ stands for a configuration with a distinguished occurrence, in this order, of the first and second component of $\vec{\mathcal{F}}_1$ (see Definition 4.15 for a more formal statement).

The rules of **BDLC**, the Hypersequent Calculus for Basic Discontinuity, are given in Figure 4.2.

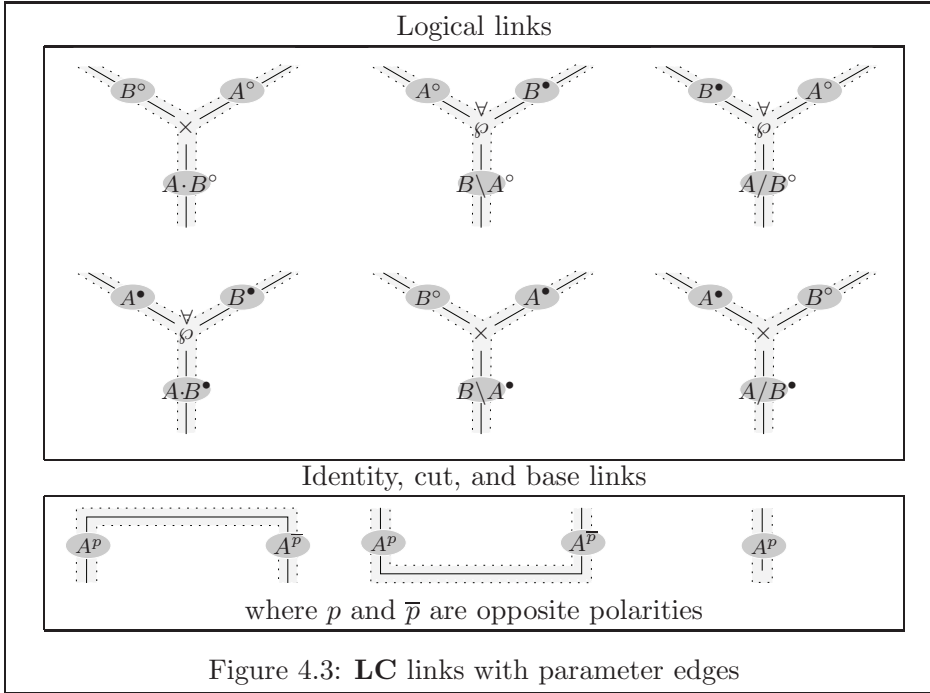
In order to develop a theory of proof nets for **BDLC**, let us focus our attention first on its continuous fragment, i.e. the Lambek Calculus **LC**.

identity and cut rules	
Id $\frac{}{A \Rightarrow A}$	Cut $\frac{\Gamma \Rightarrow A \quad \Delta[\overline{A}] \Rightarrow B}{\Delta[\Gamma] \Rightarrow B}$
logical rules for continuous operators	
·L $\frac{\Gamma[A, B] \Rightarrow C}{\Gamma[A \cdot B] \Rightarrow C}$	·R $\frac{\Gamma \Rightarrow A \quad \Delta \Rightarrow B}{\Gamma, \Delta \Rightarrow A \cdot B}$
\L $\frac{\Gamma[A] \Rightarrow C \quad \Delta \Rightarrow B}{\Gamma[\Delta, B \setminus A] \Rightarrow C}$	\R $\frac{B, \Gamma \Rightarrow A}{\Gamma \Rightarrow B \setminus A}$
/L $\frac{\Gamma[A] \Rightarrow C \quad \Delta \Rightarrow B}{\Gamma[A/B, \Delta] \Rightarrow C}$	/R $\frac{\Gamma, B \Rightarrow A}{\Gamma \Rightarrow A/B}$
logical rules for discontinuous operators	
⊙L $\frac{\Gamma[\overline{A} B] \Rightarrow C}{\Gamma[A \odot B] \Rightarrow C}$	⊙R $\frac{\Gamma \Rightarrow A \quad \Delta \Rightarrow B}{\Gamma \Delta \Rightarrow A \odot B}$
↓L $\frac{\Gamma[A] \Rightarrow C \quad \Delta \Rightarrow B}{\Gamma[\Delta B \downarrow A] \Rightarrow C}$	↓R $\frac{\overline{B} \Gamma \Rightarrow A}{\Gamma \Rightarrow B \downarrow A}$
↑L $\frac{\Gamma[A] \Rightarrow C \quad \Delta \Rightarrow B}{\Gamma[A \uparrow \overline{B} \Delta] \Rightarrow C}$	↑R $\frac{\Gamma B \Rightarrow A}{\Gamma \Rightarrow A \uparrow B}$
Figure 4.2: Rules of BDLC	

There are the usual notions of polarity and proof structures built out of links. The **BDLC** links for continuous operators are (planar) forking ribbons decorated by the usual Lambek links and with *parameter edges* being the borders of the ribbons, see Figure 4.3. The usual edges of the Lambek links are pictured as solid lines and are referred to as *predicate edges*, since the semantic reading of the proof nets are obtained from them according to the semantic trip instructions ([30, 81]). Parameter edges are pictured as dotted lines and should be thought of as syntactic trip instructions. Identity and cut links are also (non-forking) ribbons and come with parameter edges. This representation is inspired by van Benthem’s binary relational interpretation of Lambek types ([121]). Notice that, in \wp -links, one parameter edge is marked by a symbol, the quantifier \forall .⁴ Parameter edges join in the obvious way to form parameter paths and cycles. To extend all parameter paths to cycles, we add *base links* to the repertoire of links. They consist simply of parameter edges that are added at the conclusions of the proof structures.

We are ready now to rephrase in our notation Mellies’s characterization of **LC** proof nets ([67]).

⁴In this version of the theory it could be just any symbol, but in [80] the symbol really stood for a universal quantification.



Proposition 4.4 *An LC proof structure with parameter edges is correct if and only if the following conditions are satisfied:*

1. Output-uniqueness. *There is exactly one conclusion of output polarity.*
2. Danos-Regnier acyclicity. *Every predicate edge cycle crosses both edges of some \wp -link.*
3. \forall -correctness. *Every parameter cycle is \forall -correct, i.e. it contains exactly one \forall symbol.*

Observe that each predicate edge has one parameter edge on its left and one on its right. If the predicate edge is labeled by an input type, the left parameter can be thought of as the start of the type, and the right parameter as the end. For an output type this is reversed:

$$\text{start } A^\bullet \text{ end} \qquad \text{end } A^\circ \text{ start}$$

Extending the theory to discontinuity, types of sort 1 have four incident parameter edges, corresponding to a quaternary relational predication, that

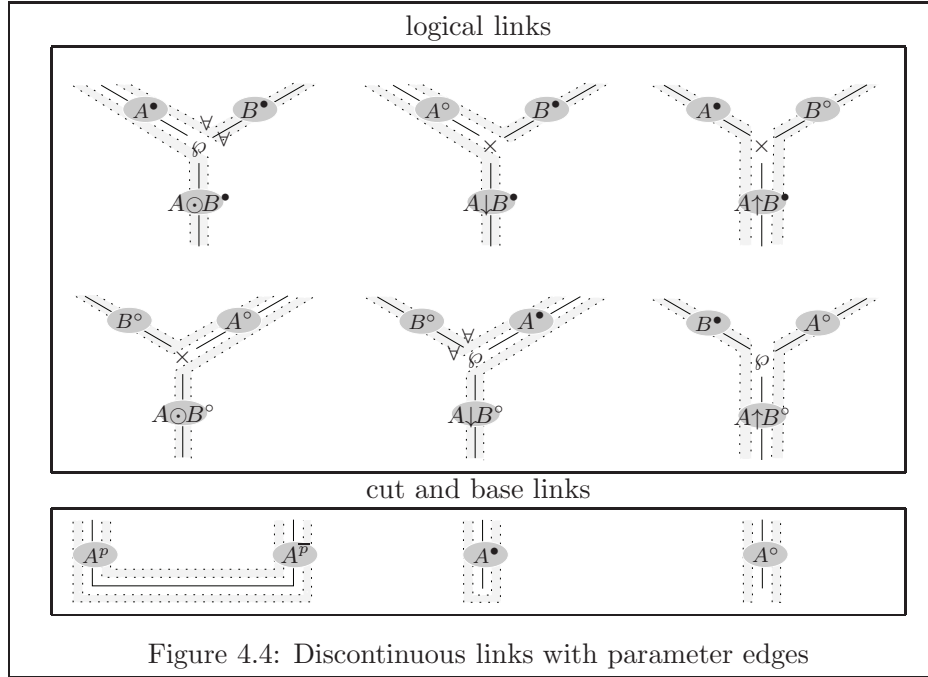


Figure 4.4: Discontinuous links with parameter edges

encodes the start and end of the two components:⁵

$$\begin{array}{l} \text{start}_1 \text{ end}_2 A^\bullet \text{ start}_2 \text{ end}_1 \\ \text{end}_1 \text{ start}_2 A^\circ \text{ end}_2 \text{ start}_1 \end{array}$$

The expanded links for the discontinuity operators are given in Figure 4.4, along with the link for cut over types of sort 1. They can no longer be regarded as ribbons. Rather parameter paths can be considered as trip instructions. There are also discontinuous base links, i.e. trip instructions on the conclusions of the proof structure that are labeled by types of sort 1. Observe that while continuous \wp -links introduce one \forall symbol each, discontinuous \wp -links either introduce no symbol or they introduce it twice.

The conditions listed in Definition 4.4 are no longer sufficient to guarantee sequentializability, as proved by the counterexample reported in Figure 4.5.

⁵With this ordering, the parameter edges do not cross inside a link. In the general theory of the next section, however, it is impossible to avoid crossing inside certain links. Parameters will be ordered as follows:

$$\begin{array}{l} \text{for any input type of sort } i: \quad \langle \text{start}_1, \text{end}_1, \text{start}_2, \text{end}_2, \dots, \text{start}_i, \text{end}_i \rangle; \\ \text{for any output type of sort } i: \quad \langle \text{end}_i, \text{start}_i, \dots, \text{end}_2, \text{start}_2, \text{end}_1, \text{start}_1 \rangle. \end{array}$$

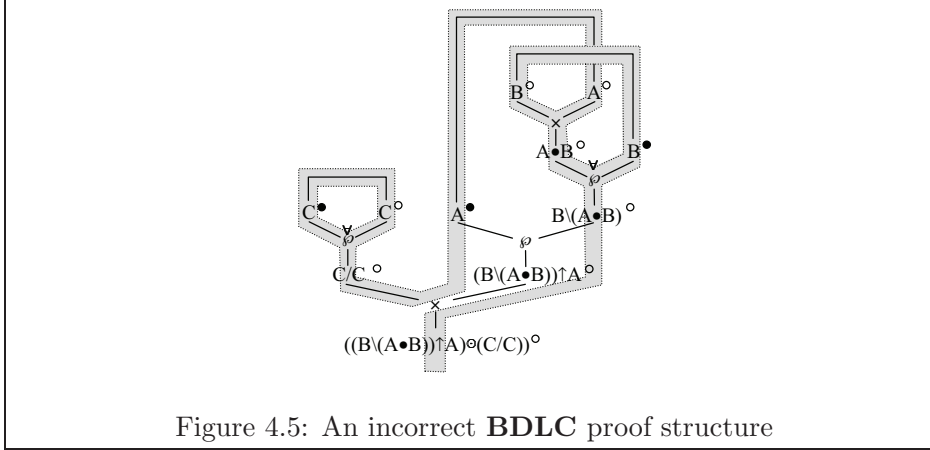


Figure 4.5: An incorrect **BDLC** proof structure

Danos-Regnier acyclicity will be replaced by the stronger condition of *hyperacyclicity*, a notion based on the concept of hyperswitching. The new condition replaces the *resolution criteria* of [80] and corrects the *input-acyclicity* condition of [83].⁶

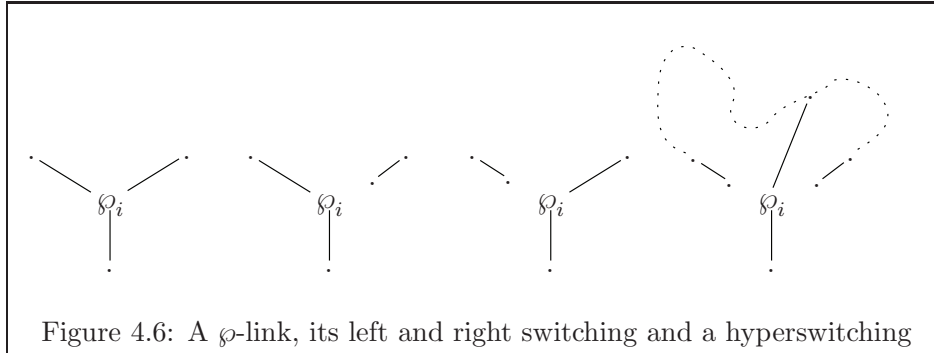
Definition 4.5 *The range of a \wp -link L in a **BDLC** proof structure Π consists of the premises of L and, if L introduces a \forall on a cycle γ , any node crossed by γ .*

Definition 4.6 *A hyperswitching of a \wp -link L is the choice of one element of the range of L . A hyperswitching s of a **BDLC** proof structure Π is the choice of a hyperswitching for each \wp -node of Π . The graph $s\Pi$ is the graph obtained from a hyperswitching s of Π disconnecting at any \wp -link its premises from the \wp -node and connecting the latter with the element of the range selected by the hyperswitching (see Figure 4.6).*

Definition 4.7 *A **BDLC** proof structure is correct, or it is a **BDLC** proof net, if:*

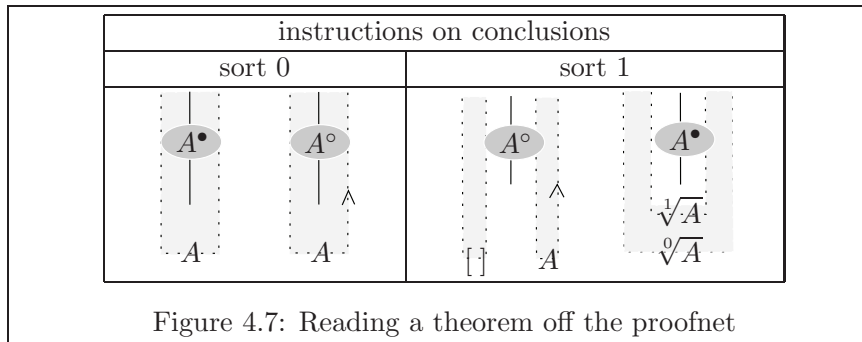
1. Output-uniqueness. *There is exactly one conclusion of output polarity.*

⁶The input-acyclicity conditions states that no parameter cycle can contain both parameter edges of a'n input type of sort zero. In general, this is too strong a condition, since it would wrongly reject, e.g., the proof structure associated to the derivation of the **BDLC** theorem $E \times (B \setminus A), ((A \uparrow B) \circ (C/C)) \setminus (E \setminus D) \Rightarrow D$. I conjecture that it is correct if we restrict the calculus to non-empty sequents.



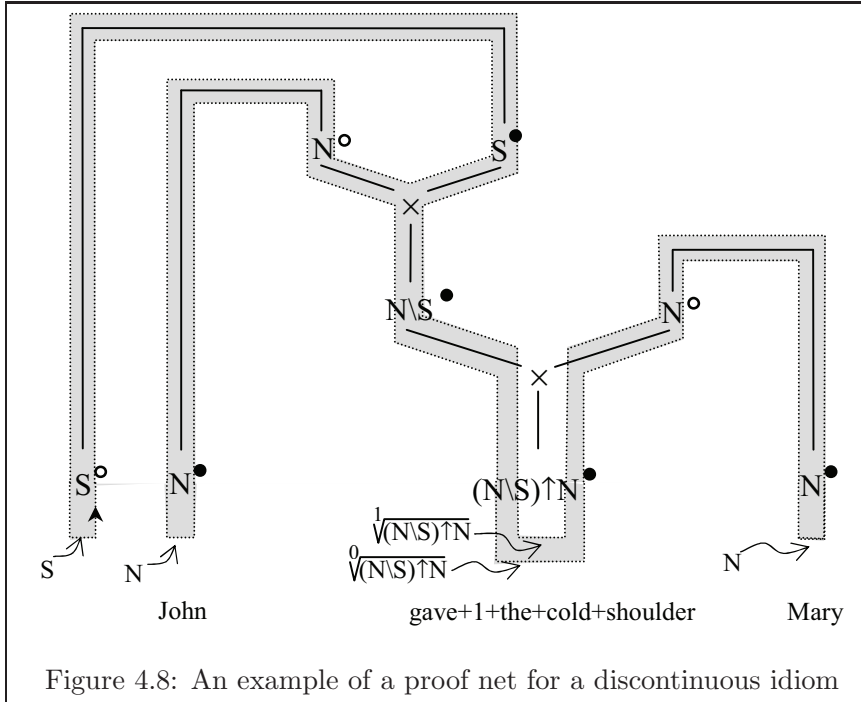
2. Hyper-acyclicity. *There is no cyclic hyperswitching.*
3. \forall -correctness. *Every parameter cycle is \forall -correct.*

As will be seen in the following sections, a **BDLC** proof net can be sequentialized as a derivation of a **BDLC** theorem, the structure of which is contained, implicitly, in the proof net. Indeed, correctness of a proof structure entails the existence of a parameter cycle –called the *prosodic trip*– that goes through all the parameter edges of the conclusions. To read the theorem off the proof net, one needs only to follow the prosodic trip starting up from the rightmost parameter edge of the unique output conclusion and write down the (components of the) successive types visited according to the instructions of Figure 4.7.



For instance, Figure 4.8 shows an example of a proof net with a discontinuous functor for the following type assignments:

gave+1+the+cold+shoulder : $(N \setminus S) \uparrow N$
John : N
Mary : N



The proof net contains a prosodic trip, from which one can read off the following sequent:⁷

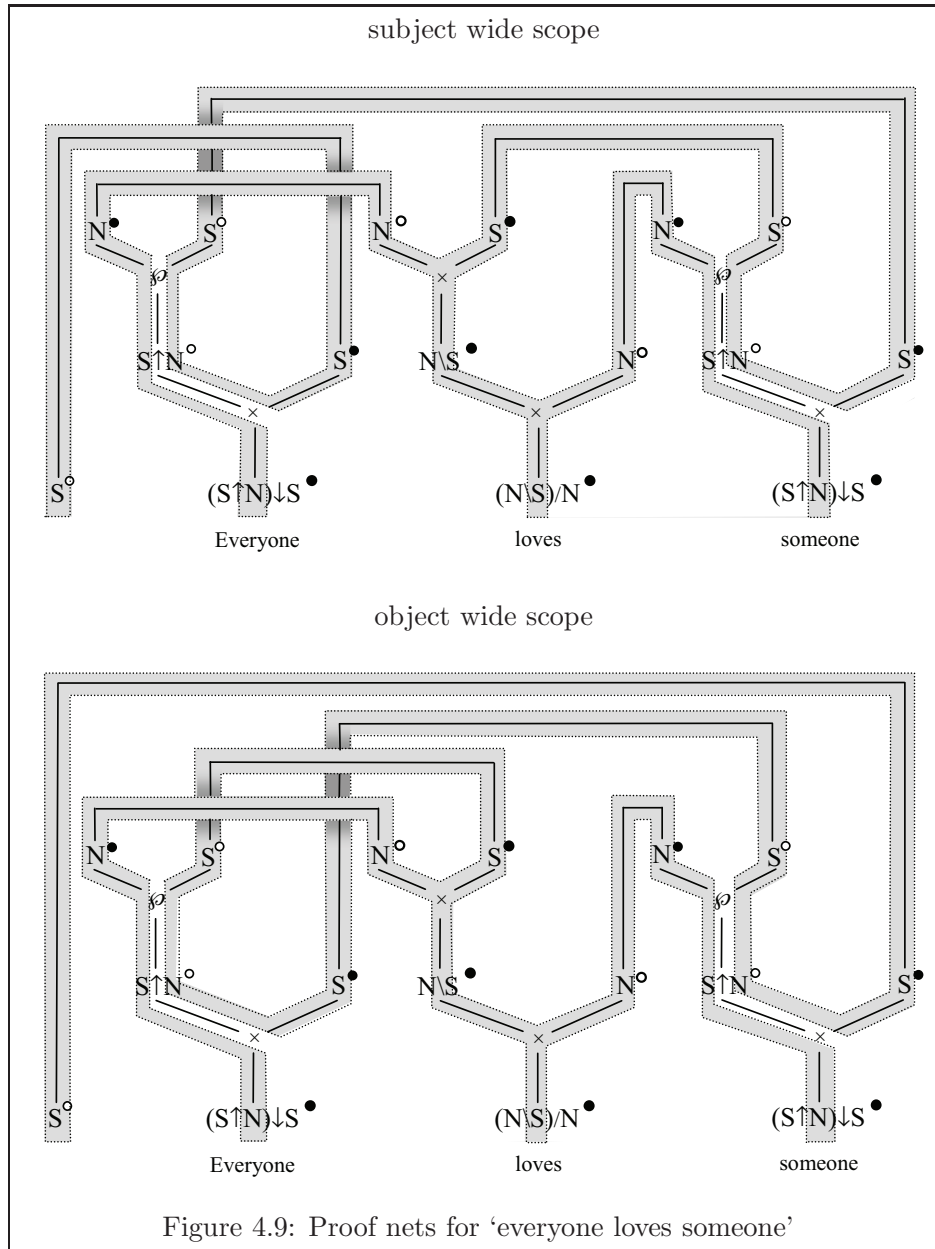
$$\begin{array}{ccccccc}
 N & , & \forall(N\S)\uparrow N & , & N & , & \forall(N\S)\uparrow N & \Rightarrow & S \\
 John & + & gave & + & Mary & + & the + cold + shower & &
 \end{array}$$

Notice that discontinuous operators are useful also in dealing with phenomena other than discontinuity. An example is the classical issue of the multiple readings of a sentence such as ‘everyone loves someone’, where the quantifiers are assigned type $(S\uparrow N)\downarrow S$. Figure 4.9 shows subject wide scope and object wide scope analyses.

4.2 The Hypersequent Calculus \mathbf{HC}_e^ω

In this section, we study the Hypersequent calculus \mathbf{HC}_e^ω and the two closely related variants $\mathbf{HC}_{\dagger_0}^\omega$ and $\mathbf{HC}_{\dagger}^\omega$. They differ only in the definition of the collection of their sequents. In particular:

⁷Since the proof net contains no \wp -link marked by a \forall symbol, all parameter edges must belong to the prosodic trip.



- the \mathbf{HC}_e^ω empty configuration of sort zero is not an $\mathbf{HC}_{t_0}^\omega$ configuration;
- the \mathbf{HC}_e^ω empty configurations of any sort are not \mathbf{HC}_f^ω configurations.

4.2.1 Types and hypersequents

Definition 4.8 For any $e \in \{\epsilon, \dagger_0, \dagger\}$, the set $\mathcal{T}_{\mathbf{HC}_e^\omega}(V)$ of \mathbf{HC}_e^ω types is the smallest set that contains, for each $h \geq 0$, the types \mathcal{F}_h of sort h generated by the following grammar on the basis of a set $V = \bigcup_{h \in \mathbb{N}} V_h$ of atomic types (where $\{V_h : h \in \mathbb{N}\}$ is a partition of V).⁸

$$\begin{aligned} \mathcal{F}_h ::= & V_h \mid \\ & (\mathcal{F}_i \cdot \mathcal{F}_j) \mid (\mathcal{F}_{i+1} \odot_l \mathcal{F}_j) \mid & (\forall i, j \in \mathbb{N} \text{ such that } i + j = h) \\ & (\mathcal{F}_k \setminus \mathcal{F}_{h+k}) \mid (\mathcal{F}_{h+k} / \mathcal{F}_k) \mid & (\forall k \in \mathbb{N}) \\ & (\mathcal{F}_k \downarrow_l \mathcal{F}_{h-1+k}) \mid (\mathcal{F}_{h-1+k} \uparrow_l \mathcal{F}_k) & (\forall k \in \mathbb{N} \text{ such that } k \leq h-1) \end{aligned}$$

where l is any positive integer less or equal to the sort of the left subtype.

A discontinuous type \mathcal{F}_h of sort h is represented, in configurations, by $h + 1$ matched occurrences. Each occurrence is referred to as a *segment* or *component*. Types of sort 0 have only one segment. The cumbersome use of coindexation can be avoided, since interleaving of segments of discontinuous types is not allowed, i.e. there is no configuration such as the following:

$$\dots, A^i, \dots, B^j, \dots, A^i, \dots, B^j, \dots$$

To achieve the matching of the components of a type of sort h it is enough to specify their relative order. Matched occurrences of an h -sorted type \mathcal{F}_h will be denoted by $\overset{\vee}{\mathcal{F}_h}, \dots, \overset{\flat}{\mathcal{F}_h}$. For any 0-sorted type \mathcal{F}_0 , the only component is $\overset{\vee}{\mathcal{F}_0}$. Clearly, $\mathcal{F}_0 = \overset{\vee}{\mathcal{F}_0}$. In a configuration of sort h there are h occurrences of the metalogical separator $[]$ marking the points of discontinuity.

Definition 4.9 The set of \mathbf{HC}_e^ω configurations is the smallest set of lists defined by the following unambiguous grammar, where Λ denotes the empty list and, for any natural number j , \mathcal{O}_j is a configuration of sort j :

$$\begin{aligned} \mathcal{O}_0 & ::= \Lambda \\ \mathcal{O}_{h+1} & ::= [], \mathcal{O}_h \\ \mathcal{O}_k & ::= \overset{\vee}{\mathcal{F}_i}, \mathcal{O}^1, \overset{\flat}{\mathcal{F}_i}, \dots, \mathcal{O}^i, \overset{\vee}{\mathcal{F}_i}, \mathcal{O}^0 \end{aligned}$$

where in the last line, for each $j \in \{0, \dots, i\}$, \mathcal{O}^j is a configuration and $k = \sum_{j=0}^i \text{sort}(\mathcal{O}^j)$.

⁸As usual, in writing a type, its outer brackets will be dropped.

As we will see below, the empty string Λ is not an $\mathbf{HC}_{\dagger_0}^\omega$ configuration. Nevertheless, we can still say, by convention, that it has sort zero.

Definition 4.10 *The set of $\mathbf{HC}_{\dagger_0}^\omega$ configurations is the smallest set of lists defined by the following unambiguous grammar where, for any natural number j , \mathcal{O}_j is a configuration of sort j :*

$$\begin{aligned}\mathcal{O}_1 & ::= [] \\ \mathcal{O}_{h+1} & ::= [], \mathcal{O}_h \\ \mathcal{O}_k & ::= \sqrt[0]{\mathcal{F}_i}, \mathcal{O}^1, \sqrt[1]{\mathcal{F}_i}, \dots, \mathcal{O}^i, \sqrt[i]{\mathcal{F}_i}, \mathcal{O}^0\end{aligned}$$

where in the last line, for each $j \in \{0, \dots, i\}$, \mathcal{O}^j is a configuration or the empty string⁹ and $k = \sum_{j=0}^i \text{sort}(\mathcal{O}^j)$.

Let Λ_h denote, for any natural number h , the list that consists of h occurrences of the metalinguistic symbol $[]$. Although in $\mathbf{HC}_{\dagger}^\omega$ no such string is a configuration, we can still set $\text{sort}(\Lambda_h)$ to be, by convention, h .

Definition 4.11 *The set of $\mathbf{HC}_{\dagger}^\omega$ configurations is the smallest set of lists defined by the following unambiguous grammar:*

$$\begin{aligned}\mathcal{O}_{h+1} & ::= [], \mathcal{O}_h \\ \mathcal{O}_k & ::= \sqrt[0]{\mathcal{F}_i}, \mathcal{O}^1, \sqrt[1]{\mathcal{F}_i}, \dots, \mathcal{O}^i, \sqrt[i]{\mathcal{F}_i}, \mathcal{O}^0\end{aligned}$$

where in the last line, for each $j \in \{0, \dots, i\}$, \mathcal{O}^j is either a configuration or a list of separators and $k = \sum_{j=0}^i \text{sort}(\mathcal{O}^j)$.

Definition 4.12 *For any $e \in \{\epsilon, \dagger_0, \dagger\}$, an \mathbf{HC}_e^ω sequent $\mathcal{O}_i \Rightarrow \mathcal{F}_i$ of sort i ($i \in \mathbb{N}$) consists of an \mathbf{HC}_e^ω configuration \mathcal{O}_i and an \mathbf{HC}_e^ω type \mathcal{F}_i , both of sort i .¹⁰*

⁹Within $\mathbf{HC}_{\dagger_0}^\omega$ it is not possible to type a configuration – such as $\sqrt[0]{\mathcal{F}_1}, \sqrt[1]{\mathcal{F}_1}$ – where two components of a type are adjacent. Nevertheless, we prefer to consider such strings as configurations, since they are needed in the versions of the hypersequent calculus that includes the bridge operator $\hat{\ } ([86, 82])$.

¹⁰We could have defined a sequent to be just a pair of a configuration and a type. Inspection of the rules, however, shows that for any provable sequent the premise configuration and the conclusion have necessarily the same sort. Thus, by restricting the language, we do not surreptitiously restrict the calculus.

It is useful to realize that \mathbf{HC}_e^ω configurations can be generated also by other grammars. For this purpose, we define *sequent wrappings* and *vector configurations*.

Definition 4.13 Let $\mathcal{O}_h = \Gamma_0, [], \dots, [], \Gamma_h$ be a list of components of \mathbf{HC}_e^ω types that contains $h \geq 1$ points of discontinuity. Let \mathcal{O} be another list of components of \mathbf{HC}_e^ω types. For any positive integer $i \leq h$, we denote by $\mathcal{O}_h|_i\mathcal{O}$ the list

$$\Gamma_0, [], \dots, \Gamma_{i-1}, \mathcal{O}, \Gamma_i, \dots, [], \Gamma_h$$

obtained wrapping \mathcal{O}_h around \mathcal{O} at the i^{th} point of discontinuity, i.e. replacing the i^{th} occurrence of $[]$ in \mathcal{O}_h by the list \mathcal{O} .

An obvious proof by induction shows that, if \mathcal{O}_h and \mathcal{O} are \mathbf{HC}_e^ω configurations, then $\mathcal{O}_h|_i\mathcal{O}$ is also an \mathbf{HC}_e^ω configuration.

Definition 4.14 Let \mathcal{F}_h be a type of sort $h \geq 1$. The vector configuration $\overrightarrow{\mathcal{F}}_h$ associated to \mathcal{F}_h is the configuration obtained separating the components of \mathcal{F}_h by occurrences of $[]$. The vector configuration $\overrightarrow{\mathcal{F}}_0$ associated to \mathcal{F}_0 is the configuration \mathcal{F}_0 of sort 0.

For instance, if $\mathcal{O}_3 = \Gamma_0, [], \Gamma_1, [], \Gamma_2, [], \Gamma_3$ and \mathcal{O} are configurations and \mathcal{F}_3 is a type, then we have:

$$\begin{aligned} \mathcal{O}_3|_1\mathcal{O} &= \Gamma_0, \mathcal{O}, \Gamma_1, [], \Gamma_2, [], \Gamma_3; \\ \overrightarrow{\mathcal{F}}_3 &= \sqrt[0]{\mathcal{F}_3}, [], \sqrt[1]{\mathcal{F}_3}, [], \sqrt[2]{\mathcal{F}_3}, [], \sqrt[3]{\mathcal{F}_3}; \\ \overrightarrow{\mathcal{F}}_3|_1\mathcal{O} &= \sqrt[0]{\mathcal{F}_3}, \mathcal{O}, \sqrt[1]{\mathcal{F}_3}, [], \sqrt[2]{\mathcal{F}_3}, [], \sqrt[3]{\mathcal{F}_3}. \end{aligned}$$

Notice that \mathbf{HC}_e^ω configurations of sort h ($h \in \mathbb{N}$) can be generated by the following grammar:

$$\mathcal{O}_h ::= \Lambda_h \mid \overrightarrow{\mathcal{F}}_h \mid \mathcal{O}_i, \mathcal{O}_j \mid \mathcal{O}_{i+1}|_l\mathcal{O}_j$$

where, in the last two cases, $i, j \in \mathbb{N}$ are such that $i + j = h$.

To obtain a grammar for $\mathbf{HC}_{\dagger_0}^\omega$, we have to modify slightly the above definition. In the first clause, h must be positive. Moreover in the last clause, \mathcal{O}_j can be either a configuration or the empty string.

For $\mathbf{HC}_{\dagger}^\omega$, on one hand we remove the clause $\mathcal{O}_h = \Lambda_h$ for any $h \geq 0$. On the other hand, in the last clause, we require that either both \mathcal{O}_{i+1} and \mathcal{O}_j are configurations, or at most one of them is an empty string of some sort.

identity and cut rules			
Id	$\overline{A \Rightarrow A}$	Cut	$\frac{\Gamma \Rightarrow A \quad \Delta[\overline{A}] \Rightarrow B}{\Delta[\Gamma] \Rightarrow B}$
logical rules ($i \in \mathbb{N}$)			
$\odot_i\text{L}$	$\frac{\Gamma[\overline{A}]_i \overline{B} \Rightarrow C}{\Gamma[A \odot_i B] \Rightarrow C}$	$\odot_i\text{R}$	$\frac{\Gamma \Rightarrow A \quad \Delta \Rightarrow B}{\Gamma _i \Delta \Rightarrow A \odot_i B}$
$\downarrow_i\text{L}$	$\frac{\Gamma[\overline{A}] \Rightarrow C \quad \Delta \Rightarrow B}{\Gamma[\Delta _i B \downarrow_i \overline{A}] \Rightarrow C}$	$\downarrow_i\text{R}$	$\frac{\overline{B} _i \Gamma \Rightarrow A}{\Gamma \Rightarrow B \downarrow_i A}$
$\uparrow_i\text{L}$	$\frac{\Gamma[\overline{A}] \Rightarrow C \quad \Delta \Rightarrow B}{\Gamma[A \uparrow_i \overline{B} _i \Delta] \Rightarrow C}$	$\uparrow_i\text{R}$	$\frac{\Gamma _i \overline{B} \Rightarrow A}{\Gamma \Rightarrow A \uparrow_i B}$

Figure 4.10: Rules of \mathbf{HC}_e^ω

4.2.2 Rules of the calculus

Let us introduce the convention whereby the unique structural operator, the comma marking concatenation, will be denoted by the (associative) operator $|_0$. The logical operators \cdot , \setminus , and $/$ will be denoted, respectively, by \odot_0 , \downarrow_0 and \uparrow_0 .

Definition 4.15 *A configuration \mathcal{O}_i of sort i appears in the context $\mathcal{O}[\]$ if $\mathcal{O}[\mathcal{O}_i]$ is a configuration that can be generated by the following grammar (where \mathcal{O} is a configuration and l is a positive integer smaller or equal to the sort of the configuration on the left of $|_l$):*

$$\mathcal{O}[\mathcal{O}_i] = \mathcal{O}_i \mid \mathcal{O}[\mathcal{O}_i]|_l \mathcal{O} \mid \mathcal{O}|_l \mathcal{O}[\mathcal{O}_i] \ .$$

The rules of \mathbf{HC}_e^ω , the hypersequent calculus for ω -discontinuity, are given in Figure 4.10.

Theorem 4.16 *If a sequent is derivable in \mathbf{HC}_e^ω then it has a Cut-free derivation.*

Proof. See the proof given by O. Valentín in the Appendix to the joint work [90]. □

Corollary 4.17 *It is decidable whether a hypersequent of \mathbf{HC}_e^ω is a theorem.*

\mathbf{HC}_e^ω	translation into \mathbf{LCP}
types	
V (atomic) $A \cdot B$ and $A \odot B$ $A/B, B \setminus A, B \downarrow_l A$, and $A \uparrow_l B$	V $f(A) \cdot f(B)$ $f(B) \multimap f(A)$
configurations	
Λ_h \overrightarrow{A} $\mathcal{O}_i, \mathcal{O}_j$ and $\mathcal{O}_i _l \mathcal{O}_j$	Λ $f(A)$ $f(\mathcal{O}_i), f(\mathcal{O}_j)$
Figure 4.11: The hyperforgetful translation f	

Proof. By backward-chaining in the finite Cut-free hypersequent search space.

The \mathbf{HC}_e^ω ($\mathbf{HC}_{\dagger_0}^\omega, \mathbf{HC}_{\dagger}^\omega$) system is a refinement of the Commutative Lambek Calculus \mathbf{LCP} with (without) the empty configuration. This is because each \mathbf{HC}_e^ω rule, translated sequent by sequent via the hyperforgetful function of Figure 4.11, yields an \mathbf{LCP} rule. Denote by $f\mathcal{D}$ the translation, sequent by sequent, of an \mathbf{HC}_e^ω derivation \mathcal{D} . The next proposition follows immediately.

Proposition 4.18 *If \mathcal{D} is an \mathbf{HC}_e^ω derivation of a sequent $\mathcal{O} \Rightarrow \mathcal{F}$, then $f\mathcal{D}$ is an \mathbf{LCP} derivation of $f\mathcal{O} \Rightarrow f\mathcal{F}$.*

This result suggests that \mathbf{HC}_e^ω proof structures might be obtained from \mathbf{LCP} proof structures, on which further restrictions are imposed. We turn now to develop this idea.

4.3 Proof nets for \mathbf{HC}_e^ω

\mathbf{HC}_e^ω proof structures are, essentially, \mathbf{LCP} proof structures constrained by deterministic trip instructions. We define the correctness criteria, and adapt Bellin and van de Wiele's results ([10]) to the context of the hypercalculus.

4.3.1 Proof structures and correctness criteria

In this section we define the notions of \mathbf{HC}_e^ω proof structure and underlying *hyperstructure*. The latter is a multigraph, in which we individuate *param-*

eter cycles. These paths are a subsidiary notion in stating the correctness criteria for \mathbf{HC}_e^ω proof structures. Moreover, any \mathbf{HC}_e^ω proof net can be sequentialized as a derivation of a \mathbf{HC}_e^ω theorem, the structure of which can be read off the proof net travelling along a special parameter cycle, called the *prosodic trip*.

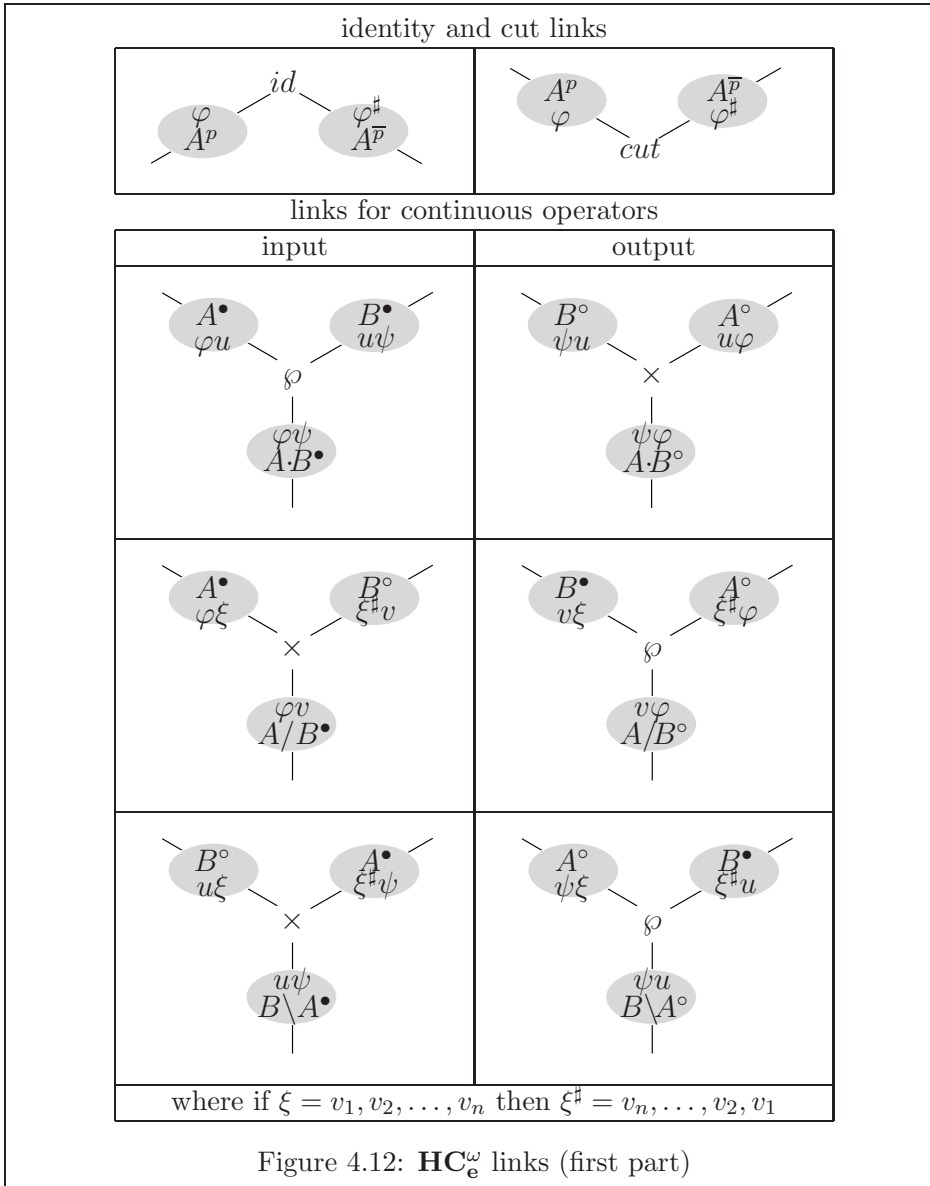
Proof structures and hyperstructures

Definition 4.19 *An \mathbf{HC}_e^ω proof structure is a connected graph such that:*

- *its edges are labeled by pairs A^p of an \mathbf{HC}_e^ω type A and a polarity symbol $p \in \{\bullet, \circ\}$;*
- *its (logical) ternary nodes are labeled either by \times or \wp ; one of the incident edges is specified to be the conclusion; the other two edges are called the premises;*
- *its (non-logical) binary nodes are labeled by ID and CUT (or more simply, represented by a horizontal line); ID nodes have two conclusions, CUT nodes have two premises;*
- *unary nodes are not labeled and have one premise;*
- *there are no other nodes;*
- *any edge is a premise of exactly one node and the conclusion of exactly one node.*

For any node N , the node together with its incident edges is called a link and N is said to be its central node. In an identity (cut) link the central node is labeled by ID (CUT). In an input (output) \circ -link the central node is ternary and the conclusion is labeled by a polarized type $A \circ B^\bullet$ ($A \circ B^\circ$), where \circ is any logical operator. In a base link, the central node is unary; its premise is called a conclusion of the proof structure. The labels of a link are related in the way explained in Figures 4.12 and 4.13 (the interpretation and use of the lists of parameters that appear on the links will be explained below).

\mathbf{HC}_e^ω links come with trip instructions encoded by lists of parameters: each incident edge labeled by a type of sort n is assigned, depending on the type of the link, a list of $2(n + 1)$ distinct integers. For typographical reasons, we write the edge label on top of the edge and the parameters of a link in between the label and the central node.



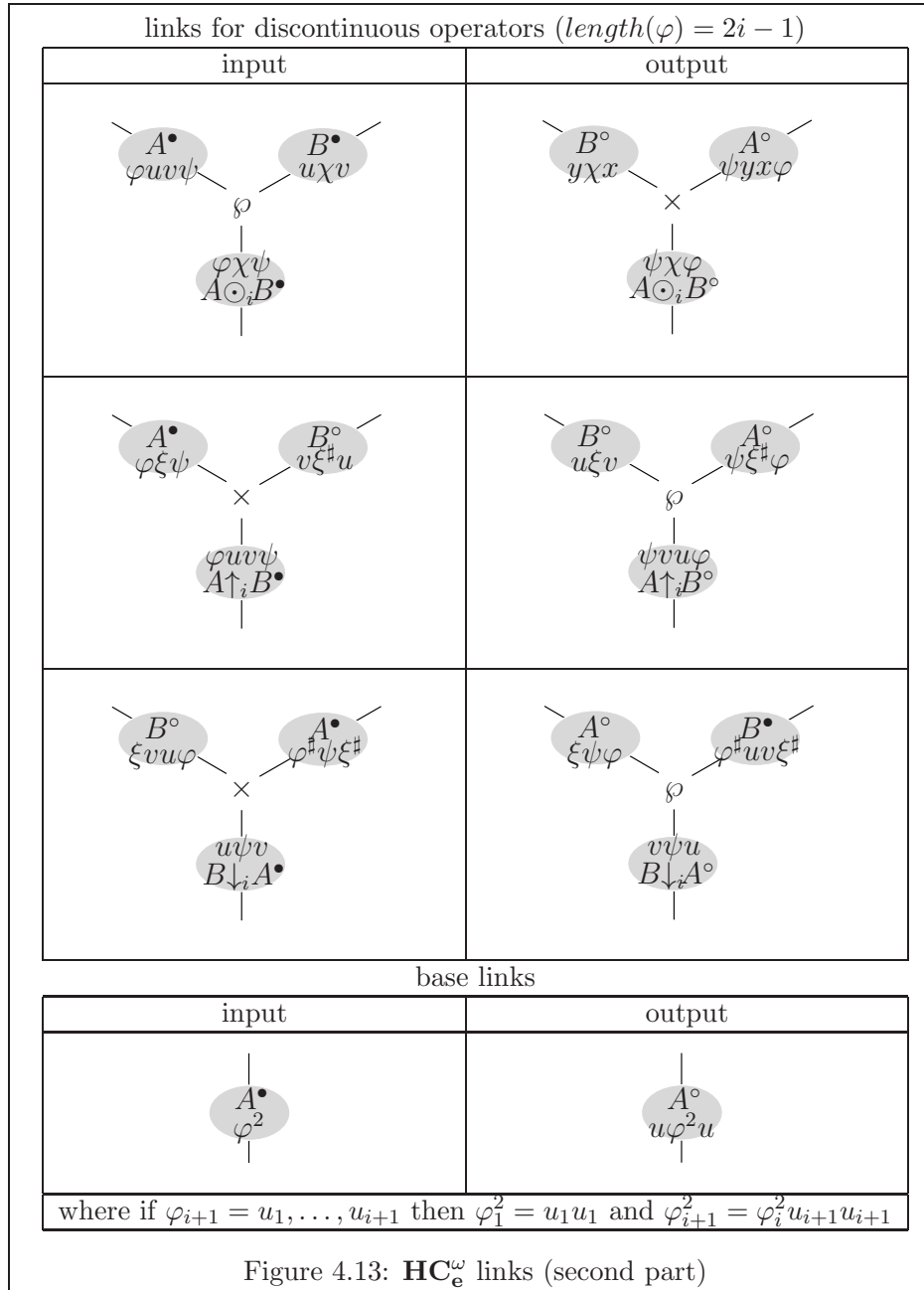
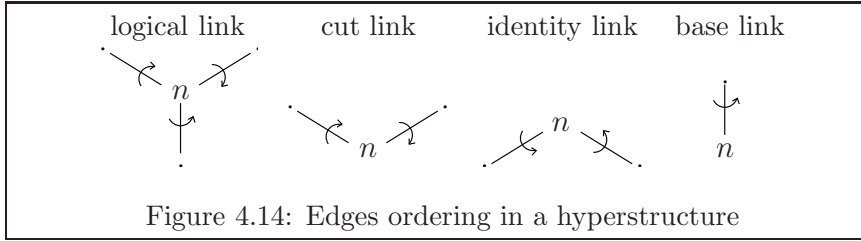


Figure 4.13: \mathbf{HC}_e^ω links (second part)



The trip instructions are a guide in identifying the parameter paths. In order to define this notion, we introduce now the concept of hyperstructure.

Definition 4.20 *The hyperstructure underlying an \mathbf{HC}_e^ω proof structure Π is the multigraph $h\Pi$ defined by the following characteristics:*

- $h\Pi$ has the same set of nodes as Π ;
- two nodes n_1 and n_2 are adjacent in $h\Pi$ if and only if they are adjacent in Π ; call $\langle n_1, n_2 \rangle$ the edge that connects them in Π ;
- any two adjacent nodes n_1 and n_2 of $h\Pi$ are connected by a bundle $b(\langle n_1, n_2 \rangle)$ of $2(n+1)$ edges, where n is the sort of the type that labels $\langle n_1, n_2 \rangle$ in Π .

Moreover, the edges of a bundle are endowed with an order. If e is a premise (conclusion) of a link with central node n , then the edges of the bundle $b(e)$ are ordered clockwise (anticlockwise) around n , as indicated in Figure 4.14.

Hyperpaths and parameter paths

The paths determined by the trip instructions in a proof structure are called *parameter paths* and are determined by *hyperpaths* in the underlying hyperstructure. We recall the notions of path in a graph and a multigraph, and then we define the notions of hyperpath and parameter path.

Definition 4.21 *A path in a graph is a list $\langle v_0, \dots, v_n \rangle$ of nodes such that for any $i \in \{1, \dots, n\}$ there is an edge e_i that connects the nodes v_{i-1} and v_i . A path is a cycle if its first and last nodes coincide. A path (cycle) is simple if its nodes are pairwise distinct (except the first and the last one).*

Definition 4.22 A path in a multigraph is a pair

$$(\langle v_0, \dots, v_n \rangle, \langle e_1, \dots, e_n \rangle)$$

of lists of nodes and edges such that for any $i \in \{1, \dots, n\}$ the edge e_i connects the nodes v_{i-1} and v_i .

Definition 4.23 Consider two edges f and g of a link L of an \mathbf{HC}_e^ω proof structure Π , where f and g are possibly the same edge. Let φ and γ be the parameters assigned to f and g by the trip instructions in L . Consider the i^{th} edge f_i of the bundle $b(f)$ and the j^{th} edge g_j of the bundle $b(g)$. We say that f_i and g_j are hypercontiguous if the i^{th} element of φ is equal to the j^{th} element of ψ .

Definition 4.24 A hyperpath in a hyperstructure $h\Pi$ is a path

$$\chi = (\langle v_0, \dots, v_n \rangle, \langle e_1, \dots, e_n \rangle)$$

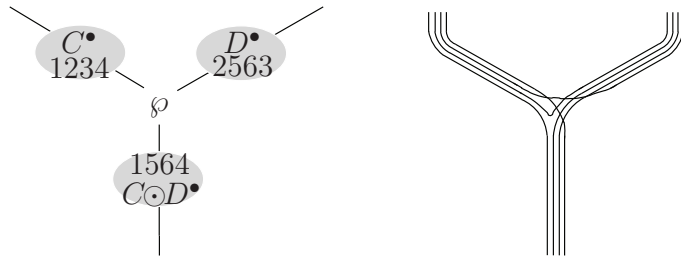
where either $n = 1$ or the following conditions hold:

- for any $i \in \{1, \dots, n-1\}$, the edges e_i and e_{i+1} are hypercontiguous;
- the edges e_1, \dots, e_n are pairwise distinct.

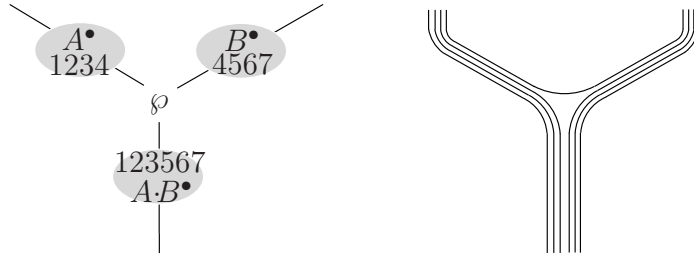
The hyperpath χ is said to underly the path $\pi = \langle v_0, \dots, v_n \rangle$ of Π . If π is a cycle, i.e. if $v_0 = v_n$, we require furthermore that e_n and e_1 are hypercontiguous. Then χ is said to be the hypercycle underlying π .

Definition 4.25 A parameter path (cycle) in an \mathbf{HC}_e^ω proof structure Π is a path (cycle) for which there is in $h\Pi$ an underlying hyperpath (hypercycle).

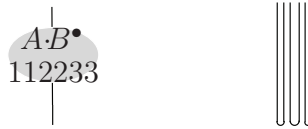
For instance, as illustrated below, there are several ways to cross –along parameter paths– a link with conclusion $C \odot D^\bullet$, where C and D are types of sort 1:



If $C \odot D^\bullet$ is the left premise of the following link, the (hyperpaths underlying the) parameter paths across the two links join in the obvious way and yield longer parameter paths:



If $A \cdot B^\bullet$ is a conclusion of the proof structure, then the parameter paths across it are joined pairwise:



Correctness criteria

The notions of *acyclic hyperswitching* and \forall -*correct* parameter cycle are the key concepts of the $\mathbf{HC}_\epsilon^\omega$ correctness criteria.

Definition 4.26 Let L be a par link in an $\mathbf{HC}_\epsilon^\omega$ proof structure Π . Let c be its central node and let n_1 and n_2 be the nodes of L connected to c by L 's premises. The range of L consists of n_1 , n_2 and any other node, different from c , that belongs to a parameter cycle supported by L . A hyperswitching of L is a choice of an element of its range.

Definition 4.27 Let s be a choice of a hyperswitching for each of the par links of a $\mathbf{HC}_\epsilon^\omega$ proof structure Π . A hyperswitching $s\Pi$ of Π is the graph obtained applying one of the following operations to any par link L of Π :

- if s selects at L one of its premises, then disconnect the other premise from L 's central node¹¹;

¹¹Formally, by disconnecting in a graph Π an edge $\langle n, m \rangle$ from the node m we mean removing the edge $\langle n, m \rangle$ and adding the edge $\langle n, l \rangle$ where l is an extra node that does not belong to Π .

- if s selects at L an element n of the range that is not a premise of L , then disconnect both premises from the central node of L and connect it with n .

A hyperswitching s of a proof structure Π is said to be cyclic if $s\Pi$ contains a cycle.

Definition 4.28 Let L be a par link in an \mathbf{HC}_e^ω proof structure Π . A parameter path (cycle) π is supported by L if two consecutive edges of a hyperpath (hypercycle) underlying π belong to the bundles of L 's premises.

Definition 4.29 Let Π be an \mathbf{HC}_e^ω proof structure. A parameter cycle that contains no conclusion of Π is \forall -correct if it is supported by exactly one par link. A parameter path/cycle that contains a conclusion of Π is \forall -correct if it is not supported by par links. A hyperpath (hypercycle) that underlies a \forall -correct parameter path (cycle) is said to be \forall -correct.

Definition 4.30 For any e in $\{\epsilon, \dagger_0, \dagger\}$, an \mathbf{HC}_e^ω proof structure is $\mathbf{HC}_\epsilon^\omega$ -correct, or is an $\mathbf{HC}_\epsilon^\omega$ proof net, if it satisfies the following correctness conditions:

1. Output-uniqueness: There is exactly one conclusion of output polarity.
2. Hyperacyclicity: There are no cyclic hyperswitchings.
3. \forall -correctness: Every parameter cycle is \forall -correct.

Observe that, because of the nature of the trip instructions, any parameter path can be extended to a parameter cycle. However, if the trip instructions on the base links are ignored, there are some maximal hyperpaths, i.e. hyperpaths that are not part of any longer hyperpath. We will refer to such hyperpaths as *prosodic paths*.

Let γ be a parameter cycle that contains some conclusion of Π . Observe that any hypercycle underlying γ consists of a list of prosodic paths. Clearly γ is \forall -correct if and only if all these prosodic paths are \forall -correct. Therefore \forall -correctness can be stated equivalently in the following way:

- every parameter cycle that does not go through conclusions of Π is \forall -correct; and
- every prosodic path is \forall -correct.

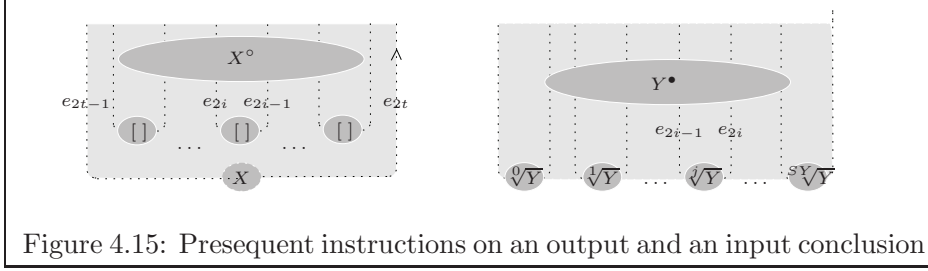


Figure 4.15: Presequent instructions on an output and an input conclusion

Prosodic trips

Some \mathbf{HC}_e^ω proof structures contain a parameter cycle, the *prosodic trip*, that goes through all the parameter edges of all the conclusions of the proof structure. Travelling along it, one can *read* –in the sense explained below– the *presequent* associated to it. This is the candidate to the sequent proved by any sequentialization of a correct proof structure.

Definition 4.31 An \mathbf{HC}_e^ω presequent $\Gamma \Rightarrow C$ is a pair comprising a \mathbf{HC}_e^ω type C and a list Γ of instances of the metalinguistic symbol $[]$ and components of \mathbf{HC}_e^ω types.

Let Π be a \mathbf{HC}_e^ω proof structure that has exactly one output conclusion X° and $n - 1 \geq 0$ input conclusions Y_1, \dots, Y_{n-1} . Let $t = \sum_{h=1}^n (Y_h + 1)$. Assume that in $h\Pi$ there is a hypercycle, defined by a sequence π of edges, that contains all the $2t$ edges of the bundles of Π 's conclusions. Let us write π so that its last two elements e_{2t-1} and e_{2t} are, in this order, the first and last edges of the bundle of the unique output conclusion. Let $\langle e_1, \dots, e_{2t} \rangle$ be the subsequence of π that contains only the edges of the bundles of the conclusions.

Let, for any polar type X^p labeling an edge e , $b(X^p)_j$ denote the j^{th} element of the bundle of e . Consider, if it exists, the sequence $\langle l_1, \dots, l_t \rangle$ where $l_t = X$ and, for any $i \in \{1, \dots, n-1\}$, l_i is defined as follows (see Figure 4.15):

- $l_i = \sqrt[j]{Y}$ if for some input conclusion Y^\bullet and $0 \leq j \leq \text{sort}(Y)$:

$$e_{2i-1} = b(Y^\bullet)_{2j+1} \text{ and } e_{2i} = b(Y^\bullet)_{2(j+1)};$$

- if $l_i = \sqrt[j]{Y}$ and $l_h = \sqrt[k]{Y}$ for some conclusion Y^\bullet and $0 \leq j, k \leq \text{sort}(Y)$:

$$i < h \text{ if and only if } j < k;$$

- $l_i = []$ if for some $h(i) \in \{1, \dots, \text{sort}(X)\}$:

$$e_{2i-1} = b(X^\circ)_{2h(i)+1} \text{ and } e_{2i} = b(X^\circ)_{2h(i)}.$$

- if $l_i = l_j = []$ for any indexes $i, j \in \{1, \dots, n-1\}$ then:

$$i < j \text{ if and only if } h(j) > h(i).$$

Definition 4.32 *If the sequence $\langle l_1, \dots, l_n \rangle$, as defined above, exists we say that π is the prosodic trip of the proof structure Π and $l_1, \dots, l_{n-1} \Rightarrow l_n$ is the presequent associated to it.*

4.3.2 Correctness and sequentialization

In this section we establish the correctness of the proof structure (\mathcal{D}) associated, in the usual way, to a derivation \mathcal{D} . We adapt Bellin and van de Wiele's characterization of empires and kingdoms, derive the Splitting Lemma in the context of \mathbf{HC}_e^ω , and finally prove some sequentialization results.

Correctness

Proposition 4.33 *If \mathcal{D} is an \mathbf{HC}_e^ω derivation of a hypersequent $\mathcal{O} \Rightarrow \mathcal{F}$, then the proof structure (\mathcal{D}) is \mathbf{HC}_e^ω -correct, it has a prosodic trip and $\mathcal{O} \Rightarrow \mathcal{F}$ is the presequent associated to it.*

Proof. The proof structure (\mathcal{D}) has, by definition, exactly one output conclusion, namely \mathcal{F}° . The other properties will be proved by induction over the length n of the derivation \mathcal{D} . If $n = 0$, the result is trivial. Suppose $n \geq 1$ and consider the last rule r in \mathcal{D} . If r introduces the type $A \cdot B$ on the left, let \mathcal{D}' be the subderivation of \mathcal{D} obtained removing r . By induction hypothesis, (\mathcal{D}') is correct and its prosodic trip π' yields a sequent $\Gamma[\vec{A}, \vec{B}] \Rightarrow C$. Adding to (\mathcal{D}') a final link L with conclusion $A \cdot B^\bullet$ breaks π' into a hypercycle γ , that goes through the last edge of $b(A)$ and the first edge of $b(B)$, and a hypercycle π that is the prosodic trip of (\mathcal{D}) and yields the presequent $\Gamma[\vec{A}, \vec{B}] \Rightarrow C$. Observe that both γ and π are \forall -correct and that no other hypercycle is modified. Therefore \forall -correctness is preserved. Moreover, any hyperswitching of (\mathcal{D}) can be obtained from a hyperswitching of (\mathcal{D}') , connecting L 's conclusion to some element of L 's range. Hence hypercyclicity is preserved. The other cases of unary rules r are similar, except

that more hypercycles might be created when inserting the link related to r . For instance, if r introduces the type $A \setminus B$ on the right, then adding to (\mathcal{D}') the link L with conclusion $A \setminus B^\circ$ splits the prosodic trip π' of (\mathcal{D}') into the prosodic trip π of (\mathcal{D}) and, because of the ordering conditions in the definition of the prosodic trip, into $2\text{sort}(A) + 1$ hypercycles, each of which goes through one edge of $b(A)$ and one of $b(B)$. Consider now the case in which the last rule r in \mathcal{D} is binary and introduces the type $A_1 \cdot A_2$. Then removing r yields, for any $i \in \{1, 2\}$, a subderivation \mathcal{D}_i of $\Gamma_i \Rightarrow A_i$. By induction hypothesis, for both i the proof structure (\mathcal{D}_i) is correct and has a prosodic trip π_i that yields the final sequent of \mathcal{D}_i . Joining (\mathcal{D}_1) and (\mathcal{D}_2) via the final link L with conclusion $A_1 \cdot A_2^\circ$ results in creating (\mathcal{D}) and in fusing π_1 and π_2 into the prosodic trip π of (\mathcal{D}) . Observe that π is \forall -correct and that any other hyperpath of $h(\mathcal{D})$ is contained either in $h(\mathcal{D}_1)$ or in $h(\mathcal{D}_2)$. As a consequence, no hypercycle supported by a par link spans over the two substructures. Therefore (\mathcal{D}) is \forall -correct and has no cyclic hyperswitchings. All the other binary cases, including the cut rule, are similar. \square

Empires and kingdoms

Since proof structures are graphs and graphs are defined on the bases of their sets of nodes and edges, it makes sense to compare them using the subset relation. For instance, a substructure \mathcal{S} of an \mathbf{HC}_e^ω proof structure Π is a proof structure the nodes and edges of which are all nodes and edges of Π . For the same reason, it makes sense to combine (non-disjoint) substructures with unions and intersections. In particular if A labels an edge of an \mathbf{HC}_e^ω proof net Π , one can consider the partially ordered set $\mathcal{S}(A)$ of subnets of Π that contain A among their conclusions. We show now that the poset $\mathcal{S}(A)$ is a lattice and we characterize its maximal element, the *empire* of A , thereby proving that $\mathcal{S}(A)$ is not empty.

Definition 4.34 *A subnet of a proof net Π is a substructure \mathcal{S} of Π that is a proof net. The empire (kingdom) of an edge A of Π is, if it exists, the biggest (smallest) subnet eA (kA) of Π in which A is a conclusion.*

Proposition 4.35 *If two subnets of an \mathbf{HC}_e^ω proofnet share at least an edge, then their intersection is a subnet.*

Proof. Let \mathcal{S} and \mathcal{T} be subnets of an \mathbf{HC}_e^ω proofnet that share at least an edge. Their intersection \mathcal{R} is clearly a proof structure. It is \forall -correct because, on the one hand, any proper hypercycle of $h\mathcal{R}$ is a proper hypercycle

of $h\Pi$ and therefore it is supported by exactly one par link. On the other hand, any prosodic trip τ of \mathcal{R} is part of either a prosodic trip of Π or a hypercycle that is supported by a par link the conclusion of which does not lie in \mathcal{R} . In either case, τ is not supported by any par link in \mathcal{R} . Moreover \mathcal{R} has no cyclic hyperswitching. Indeed, if it had a cyclic hyperswitching then it could be extended to a hyperswitching of, say, \mathcal{S} . The extension would still be cyclic, in contradiction with the correctness of \mathcal{S} . Finally, \mathcal{R} has exactly one output conclusion, because $f\mathcal{R} = f\mathcal{S} \cap f\mathcal{T}$ and the latter has one output conclusion, since any of its switchings is connected and acyclic because the set of proof nets of linear logic is closed under non-empty intersection. \square

Proposition 4.36 *If two subnets of an \mathbf{HC}_e^ω proof net are not disjoint, then their union is a subnet.*

Proof. Let \mathcal{R} be the union of two non-disjoint subnets \mathcal{S} and \mathcal{T} . \mathcal{R} is \forall -correct. Indeed, on the one hand, any proper parameter cycle in \mathcal{R} lies either in \mathcal{S} or in \mathcal{T} and therefore is supported by exactly one par link. On the other hand any prosodic path in \mathcal{R} extends either to a prosodic path of Π or to a proper parameter path in Π . In the former case, the prosodic trip is supported by no par link in Π and, a fortiori, in \mathcal{R} . In the latter case the supporting par link cannot lie in \mathcal{R} , for otherwise the whole parameter cycle would be contained therein, and therefore does not support the parameter path in kC . Hyperacyclicity is inherited from Π , since any hyperswitching of kC can be extended to a hyperswitching of Π . Output-uniqueness follows from connectedness of all switchings of $h\mathcal{R}$, i.e. connectedness of all simple hyperswitchings of \mathcal{R} . The latter property holds since any simple hyperswitching of \mathcal{R} , when restricted to \mathcal{S} or \mathcal{T} , is connected and because \mathcal{S} and \mathcal{T} are non-disjoint. \square

To prove that $\mathcal{S}(A)$ is not empty we characterize the empire of A , via the notion of *trimming*.

Definition 4.37 *Consider a premise A of a link L in an \mathbf{HC}_e^ω proof structure Π . For any switching s of Π , the trimming of $s\Pi$ at A is the graph $s\Pi_A$ defined as follows:*

- if L is a conclusion link or a par link at which σ does not select A , let $s\Pi_A = s\Pi$;

- if L is a par link at which s selects A or it is a tensor or cut link, disconnect A from the central node of L and let $s\Pi_A$ be the component that contains A .

Switchings and trimmings can be compared, qua graphs, by the subset relation and combined with the operations of union and intersection.

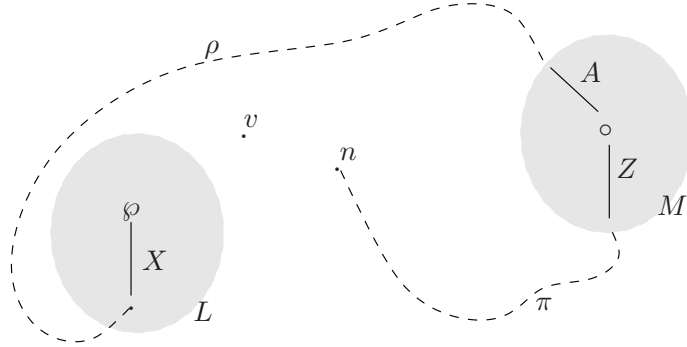
Theorem 4.38 *Let Π be an $\mathbf{HC}_\epsilon^\omega$ proof net. For any edge A of Π , the empire eA of A in Π exists and is the proof net determined in the following equivalent ways:*

1. $\mathcal{S} = \bigcap_s s\Pi_A$ where s varies over all hyperswitchings of Π ;
2. the smallest substructure \mathcal{T} that contains A and is closed under the following conditions on links, the premises of which do not include A :
 - (a) if one premise of a tensor or cut link is in \mathcal{T} , then the link is in \mathcal{T} ;
 - (b) the range of a \wp link is in \mathcal{T} if and only if the link is in \mathcal{T} .

Proof. Let \mathcal{K} be the set of substructures of Π that contain A and are closed under conditions (a) and (b) on links, the premises of which do not include A . Observe that \mathcal{S} is an element of \mathcal{K} (Lemma 4.39) and is minimal therein (Lemma 4.40). Therefore \mathcal{T} exists and coincides with \mathcal{S} . Moreover, \mathcal{S} is $\mathbf{HC}_\epsilon^\omega$ -correct (Lemma 4.41, 4.42, and 4.43) and is maximal in $\mathcal{S}(A)$ (Lemma 4.44). \square

Lemma 4.39 *\mathcal{S} is an element of \mathcal{K} .*

Proof. That \mathcal{S} is a substructure that contains A and satisfies condition (a) is immediate. As for one implication of (b), suppose that the range of some \wp link L is in \mathcal{S} . If L 's conclusion X is not in \mathcal{S} , then A is a premise of a logical link with conclusion Z (or of a cut link, the other premise of which is Z) and in some hyperswitching s there is a path $\langle X, \pi, Z, A \rangle$. But L is switched to elements of L 's range and therefore X and A are also part of another path contained in s . Hence s is proved to be cyclic in contradiction with Π 's hyperacyclicity. Then L 's conclusion must be in \mathcal{S} as well. As for the other implication of (b), assume that the premises and the conclusion X of a \wp link L is in \mathcal{S} (the following reasoning is illustrated in the picture below).



Suppose some node n of L 's range is not in \mathcal{S} . Then there is a hyperswitching s such that X , but not n , is in $s\Pi_A$. Hence, A is a premise of a logical (or cut) link M , the conclusion (other premise) of which we call Z . Moreover, if M is a \wp link, then s chooses A at M . Let $\langle \pi, Z \rangle$ be the path in $s\Pi$ the first node of which is n . Clearly $\langle \pi, Z \rangle$ and $s\Pi_A$ do not share any edge. The hyperswitching s does not choose n at L , for otherwise s would be cyclic. It chooses some other element v of the range. The edge E that joins L 's central node with v is part of $s\Pi_A$ and therefore it does not belong to $\langle \pi, Z \rangle$. Consider the switching t that is just like s except for choosing n at L . Since X is in \mathcal{S} , there is a path $\langle X, \rho, A \rangle$ in $t\Pi_A$ that does not contain Z . But $\langle \pi, Z \rangle$ is also contained in $t\Pi$, which therefore contains a cycle. Since this is in contradiction with Π 's hyperacyclicity, the range of L must be contained in \mathcal{S} . It is thus established that \mathcal{S} belongs to the set \mathcal{K} . \square

Lemma 4.40 \mathcal{S} is the smallest element of \mathcal{K} .

Proof. Observe that \mathcal{K} is closed under intersection. Therefore \mathcal{K} contains a minimal element \mathcal{T} . Suppose that $\mathcal{T} \subsetneq \mathcal{S}$ and let $W \notin \mathcal{T}$ be an edge in \mathcal{S} . Consider a hyperswitching s such that:

- α) if A is a premise of a \wp link L , then s selects A at L ; otherwise,
- β) if a premise, but not the conclusion, of a \wp link L is in \mathcal{T} , then s selects at L an element of its range that is not in \mathcal{T} .

Since W is in \mathcal{S} , there is a path π in $s\Pi_A$ that starts with A (that is in \mathcal{T}) and ends with W (that is not in \mathcal{T}). Since \mathcal{T} is a substructure that has properties (a) and (b), the path π can only exit \mathcal{T} passing through a premise X of a link L such that X , but not L 's conclusion, is in \mathcal{T} and L is one of the following links:

- (i) a tensor link, a premise of which is A ;
- (ii) a \wp link, a premise of which is A ;
- (iii) a \wp link such that \mathcal{T} contains one of its premises but not all its range.

But none of these options are available, respectively, for the following reasons:

- (i) if $X = A$ since $\pi \subseteq s\Pi_A$ and if $X \neq A$ because s is acyclic;
- (ii) s would select A at L and $\pi \subseteq s\Pi_A$;
- (iii) by construction of s .

Therefore $\mathcal{S} = \mathcal{T}$. □

Lemma 4.41 *Every parameter cycle in \mathcal{S} is \forall -correct.*

Proof. Consider a hypercycle γ of $h\mathcal{S}$. There are two possibilities. First, suppose that γ goes through no conclusion of \mathcal{S} or it goes only through conclusions of \mathcal{S} that are also conclusions of Π . Then γ is a hypercycle in $h\Pi$ and therefore it is \forall -correct. Secondly, suppose that γ goes through some conclusion of \mathcal{S} that is not a conclusion of Π . Then γ , as a path of $h\Pi$, splits into a number of hyperpaths $\gamma_1, \dots, \gamma_n$ that are each part of a hypercycle $\delta_1, \dots, \delta_n$ of $h\Pi$ (where $\delta_1, \dots, \delta_n$ are not necessarily distinct hypercycles). If, for some $i \in \{1, \dots, n\}$, δ_i goes through some conclusion of Π , then δ_i and, a fortiori, γ_i is \forall -correct. If, for some $i \in \{1, \dots, n\}$, δ_i does not go through some conclusion of Π , then it must be supported by exactly one par link L . Because of property (b) the conclusion of L cannot be part of \mathcal{S} because δ_i , and therefore the range of L , is not all in \mathcal{S} . Hence γ_i is \forall -correct. Since $\gamma_1, \dots, \gamma_n$ are all \forall -correct, so is γ . □

Lemma 4.42 *Every hyperswitching of \mathcal{S} is acyclic.*

Proof. Notice that, because of property (b), no par link of \mathcal{S} can be switched to an element outside of the substructure. Hence any hyperswitching s of \mathcal{S} can be extended to a hyperswitching of Π and must therefore be acyclic. □

Lemma 4.43 \mathcal{S} has exactly one output conclusion.

Proof. Observe that \mathcal{S} has the same number of output conclusions as $f\mathcal{S}$, where f is the hyperforgetful function defined in Figure 4.11. Therefore it is enough to show that $f\mathcal{S}$ has exactly one output conclusion. Notice that $f\Pi$ has no cyclic switchings and has exactly one output conclusion. Therefore, by Theorem 2.38, all of its switchings are connected graphs. Then any hyperswitching of Π is a connected graph. Indeed, assume that a hyperswitching s has more than one component. Suppose that a par link L of $s\Pi$ is switched to an element n of its range that is not any of its premises. Consider a hyperswitching t that is just like s except for choosing at L one of its premises. Observe that t and s have the same number of components. This is because disconnecting L 's conclusion from n increases the number of components by one (for otherwise t would have been cyclic) and connecting it with the chosen premise reduces the number of components by one (for otherwise s would be cyclic). Iterating this reasoning we can obtain a simple hyperswitching of Π that has the same number of components as s . But s can have only one component, because its choice of premises of par links determines also a switching of $f\Pi$. Hence any hyperswitching t of Π must be connected. Any simple hyperswitching of \mathcal{S} can be extended to a hyperswitching t of Π that respects conditions $\alpha)$ and $\beta)$ of Lemma 4.40. Now, $t\Pi$ is connected and therefore $s = t\Pi_A$ is connected. Therefore there is a connected switching of $f\mathcal{S}$ which, by Theorem 2.38, must have exactly one output conclusion. \square

Lemma 4.44 \mathcal{S} is the empire of A in Π .

Proof. Let \mathcal{R} be a subnet of Π that has A among its conclusions and suppose that there is an edge Z that belongs to \mathcal{R} but not to \mathcal{S} . Hence Z is not part of a trimming $s\Pi_A$ for some hyperswitching s of Π . Since \mathcal{R} is \forall -correct, no par link of \mathcal{R} is switched to an element outside of \mathcal{R} and therefore it is possible to restrict s to a hyperswitching t of \mathcal{R} . Since every hyperswitching of \mathcal{R} is connected, Z and A have to be part of a path in $t\mathcal{R}$. But this cannot be the case since A is a conclusion of \mathcal{R} and $Z \notin s\Pi_A$. \square

Since $\mathcal{S}(A)$ is not empty and is closed under intersections it has a minimal element, the kingdom of A .

The Splitting Lemma

An $\mathbf{HC}_\epsilon^\omega$ proof net is in splitting conditions if it has no final par link, and at least a tensor or cut link. The *Splitting Lemma* shows that any proof net that satisfies these conditions contains a final link L the removal of which results in splitting the proof net into the (disjoint) empires of L 's premises. To establish this result, some technical lemmas are needed.

Definition 4.45 *Let P and Q be the premises in an $\mathbf{HC}_\epsilon^\omega$ proof net of a cut link L with central node C . The kingdom kC of the cut link L is the graph obtained joining the kingdoms kP and kQ by the cut link L .*

Proposition 4.46 *The kingdom kC of a cut link L in an $\mathbf{HC}_\epsilon^\omega$ proof net Π is the smallest subnet of Π that contains L .*

Proof. Clearly, kC is a substructure. It is \forall -correct. Indeed, on the one hand, any proper parameter cycle in kC lies either in kP or in kQ and therefore is supported by exactly one par link. On the other hand any prosodic path in kC extends either to a prosodic path of Π or to a proper parameter path in Π . In the former case, the prosodic trip is supported by no par link in Π and, a fortiori, in kC . In the latter case the supporting par link cannot lie in kC , for otherwise the whole parameter cycle would be contained therein, and therefore would not support the parameter path in kC . Hypercyclicity is inherited from Π , since any hyperswitching of kC can be extended to a hyperswitching of Π . Finally, kC has a unique output conclusion, because the polarity of its premises are opposite and both kP and kQ have exactly one output conclusion. \square

The above reasoning establishes the following result.

Proposition 4.47 *The kingdom kC of the conclusion C of a tensor link L is the result of joining the kingdoms of L 's premises by the link L .*

To unify the previous two results in a unique statement, we will hereafter refer to the central node of a cut link L as its conclusion.

Lemma 4.48 *For any $i \in \{0, 1\}$, let P_i be a premise of a link L_i in an $\mathbf{HC}_\epsilon^\omega$ proof net, and let C_i be L_i 's conclusion, if L_i is a logical link, or L_i 's central node if L_i is a cut link. Suppose that $C_0 \neq C_1$ and $P_1 \in eP_0$. Then:*

$$C_1 \notin eP_0 \text{ if and only if } C_0 \in kC_1.$$

Proof. Observe that, since $P_1 \in eP_0 \cap kC_1$ then both $\mathcal{R} = eP_0 \cap kC_1$ and $\mathcal{S} = eP_0 \cup kC_1$ are subnets. Assume that $C_1 \notin eP_0$. Then eP_0 is properly contained in \mathcal{S} . If $C_0 \notin kC_1$, then \mathcal{S} would be a subnet of Π that has P_0 among its conclusions, thus contradicting eP_0 's maximality among such subnets. Therefore $C_0 \in kC_1$. To prove the other implication, assume now that $C_0 \in kC_1$. Then \mathcal{R} is properly contained in kC_1 . If $C_1 \notin eP_0$, then \mathcal{R} would be a subnet of Π that has C_1 among its conclusions, thus contradicting kC_1 's minimality among such subnets. Therefore $C_1 \in eP_0$. \square

Let $\mathcal{C}\Pi$ be the set of the conclusions of logical links and the central nodes of cut links in an $\mathbf{HC}_\epsilon^\omega$ proof net. Consider the relation \ll defined as follows on $\mathcal{C}\Pi$:

$$C_0 \ll C_1 \text{ if and only if } C_0 \in kC_1.$$

Lemma 4.49 *Let Π be an $\mathbf{HC}_\epsilon^\omega$ proof net and let C_0 and C_1 be elements of $\mathcal{C}\Pi$. If $C_0 \ll C_1$ and $C_1 \ll C_0$, then C_0 and C_1 coincide.*

Proof. Suppose that C_0 and C_1 do not coincide. Then $kC_0 = kC_0 \cap kC_1 = kC_1$. If C_0 is the conclusion of a par link L , removing it yields a smaller subnet, contradicting kC_1 's minimality. If C_0 is the conclusion of a tensor link L (or the central node of a cut link L), then kC_0 is the result of joining the kingdoms of L 's premises by the link L . Then C_1 belongs to the kingdom kP_0 of a premise P_0 of L and therefore it belongs to the empire eP_0 . But by Lemma 4.48, $C_1 \notin eP_0$. Hence C_0 and C_1 coincide. \square

Lemma 4.50 (The Splitting Lemma) *Any $\mathbf{HC}_\epsilon^\omega$ proof net Π in splitting conditions has a splitting tensor or cut link.*

Proof. Consider an element C_0 of $\mathcal{C}\Pi$ that is maximal with respect to \ll . Let A_0 and A_1 be the premises of the link L with conclusion C_0 . Then L is a splitting tensor or cut link. Clearly, only L 's central node belongs to $eA_0 \cap eA_1$. Moreover, Π can be obtained joining eA_0 and eA_1 with L . Indeed, if this was not the case, then for some $i \in \{0, 1\}$ there would be a premise $P_1 \in eA_i$ of a link with conclusion $C_1 \notin eA_i$ such that C_1 is at or above an element D of $\mathcal{C}\Pi$. By Lemma 4.48, $C_0 \in kC_1$. Since $kC_1 \subseteq kD$, then $C_0 \in kD$, i.e. $C_0 \ll D$, which is in contradiction with C_0 's maximality. Therefore eA_0 and eA_1 joined by L yield Π . \square

Sequentialization

The proof of the Sequentialization Theorem is based on a reasoning by induction over the complexity of the proof net. One way to reduce the complexity is to apply the Splitting Lemma. Another way is to remove a final \wp -link. There might be more ways to sequentialize a proof net, but their differences are not substantial.

Lemma 4.51 *Removal of a final \wp -link from an $\mathbf{HC}_\epsilon^\omega$ proof net preserves correctness.*

Proof. Removal of a final \wp link L preserves each correctness condition for the following reasons:

- output-uniqueness: in any \wp -link there are as many output premises as output conclusions;
- hyperacyclicity: the existence of cycles does not depend on a final par link;
- \forall -correctness of proper parameter cycles: after L 's removal there are the same or fewer parameter cycles;
- \forall -correctness of prosodic paths: some prosodic paths are shortened removing final parameter edges and, possibly, some parameter cycles turn into prosodic paths since the edges of the bundles of L 's premises are removed.

□

Theorem 4.52 (Sequentialization) *Any $\mathbf{HC}_\epsilon^\omega$ proof net Π has a prosodic trip τ and there is an $\mathbf{HC}_\epsilon^\omega$ derivation \mathcal{D} of the hypersequent $\mathcal{O} \Rightarrow \mathcal{F}$ associated to τ such that $(\mathcal{D}) = \Pi$.*

Proof. The proof is by induction over the complexity of Π . If Π contains only an identity link with conclusion A^\bullet and A° , then Π has a prosodic trip and it yields the sequent $A \Rightarrow A$. Suppose now that Π has a final par link L . Removing L yields a proof structure Π' that, by Lemma 4.51, is $\mathbf{HC}_\epsilon^\omega$ -correct. By induction hypothesis, Π' can be sequentialized as a derivation \mathcal{D}' of the sequent associated to the prosodic trip τ' of Π' . Observe that if two

edges e_1 and e_2 of L 's premises bundles are part of a hypercycle γ_e in $h\Pi$, then γ_e is equal to $\langle e_1, e_2, \delta_e \rangle$ for some hyperpath δ_e that, because of \forall -correctness, contains no edge of the bundles of either Π 's conclusions or L 's premises. Such edges e_1 and e_2 bear in L the same parameter and determine the relative position of L 's premises in the hypersequent associated to τ' . All possible cases are reviewed in the following table. Then the derivation \mathcal{D}' can be completed to a derivation \mathcal{D} , simply adding the rule that introduces the conclusion of L .

L 's conclusion	L 's premises	hypersequent associated to τ'	rule to be added to \mathcal{D}'
$A \cdot B^\bullet$	A^\bullet B^\bullet φu $w\psi$	$\Gamma[\vec{A}, \vec{B}] \Rightarrow C$	left \cdot
$A \odot_i B^\bullet$	A^\bullet B^\bullet $\varphi uv\psi$ $u\chi v$	$\Gamma[\vec{A} _i \vec{B}] \Rightarrow C$	left \odot_i
A/B°	B^\bullet A° $v\xi$ $\xi^\# \varphi$	$\Gamma, \vec{B} \Rightarrow A$	right $/$
$B \setminus A^\circ$	A° B^\bullet $\varphi\xi$ $\xi^\# u$	$\vec{B}, \Gamma \Rightarrow A$	right \setminus
$A \uparrow_i B^\circ$	B^\bullet A° $u\xi v$ $\psi\xi^\# \varphi$	$\Gamma _i \vec{B} \Rightarrow A$	right \uparrow_i
$B \downarrow_i A^\circ$	A° B^\bullet $\psi\chi\varphi$ $\varphi^\# uv\psi^\#$	$\vec{B} _i \Gamma \Rightarrow A$	right \downarrow_i

If Π is in splitting conditions then by Lemma 4.50 the proof net Π splits, removing a cut or final tensor link L , into two subnets Π_1 and Π_2 . By induction hypothesis, for any $i = 0, 1$ there is a derivation \mathcal{D}_i of the sequent associated to the prosodic trip τ_i of Π_i . In this case, the edges of L 's premises bundles that bear identical parameters belong to prosodic paths and not to proper parameter cycles, for otherwise one of the Π_i 's would not be correct. Rule by rule inspection shows that the prosodic paths contain enough information to guarantee that the derivations \mathcal{D}_1 and \mathcal{D}_2 can be combined by the binary rule that introduces L 's conclusion (or that performs the cut on L 's premises, if L is a cut link) to form a derivation \mathcal{D} and that $(\mathcal{D}) = \Pi$ has a prosodic trip that yields the sequent proved by \mathcal{D} . \square

Corollary 4.53 *An $\mathbf{HC}_\epsilon^\omega$ proof structure is correct if and only if it is sequentializable in $\mathbf{HC}_\epsilon^\omega$.*

Two $\mathbf{HC}_\epsilon^\omega$ derivations \mathcal{D} and \mathcal{D}' of a hypersequent are \equiv -equivalent if there is a sequence of $\mathbf{HC}_\epsilon^\omega$ derivations $\mathcal{D} = \mathcal{D}_1, \dots, \mathcal{D}_n = \mathcal{D}'$ such that, for any $i \in \{1, \dots, n-1\}$, \mathcal{D}_i and \mathcal{D}_{i+1} differ at most in the order of application of two consecutive inferences.

Theorem 4.54 *Let \mathcal{D} and \mathcal{D}' be $\mathbf{HC}_\epsilon^\omega$ derivations of the same hypersequent. If $(\mathcal{D}) = (\mathcal{D}')$ then $\mathcal{D} \equiv \mathcal{D}'$.*

Proof. The proof is by induction over $n(\mathcal{D}, \mathcal{D}') = \sum_\gamma k(\gamma)$ where γ ranges over all branches of the derivation \mathcal{D} and $k(\gamma)$ is defined as follows. For any branch γ of \mathcal{D} , let I_γ be the last (bottom up) rule where \mathcal{D} and \mathcal{D}' agree. If I_γ is an identity link, then \mathcal{D} and \mathcal{D}' agree in the order of application of rules along this branch and we set $k(\gamma) = 0$. If I_γ is not an identity link, let $k(\gamma)$ be the number of rules that precede I_γ in γ . If $n(\mathcal{D}, \mathcal{D}') = 0$ then $\mathcal{D} = \mathcal{D}'$ and there is nothing to show. Suppose instead that for some branch γ , $k(\gamma) \geq 1$. Let A be the active type of the inference I_A above I_γ in γ . Let I'_A be the rule in \mathcal{D}' , in which the same occurrence of A is active. Let I'_1, \dots, I'_k be the rules in \mathcal{D}' that intervene, in this order, between I'_A and I_γ . We will show, by induction over k , that there are derivations $\mathcal{D}_0 = \mathcal{D}', \mathcal{D}_1, \dots, \mathcal{D}_k$ such that for all $i = 1, \dots, k$ the derivation \mathcal{D}_i is obtained from \mathcal{D}_{i-1} permuting I'_A below I'_i . Clearly, $\mathcal{D}_k \equiv \mathcal{D}'$. Since $n(\mathcal{D}, \mathcal{D}_k) = n(\mathcal{D}, \mathcal{D}') - 1$, by induction hypothesis $\mathcal{D} \equiv \mathcal{D}_k$. Thus $\mathcal{D} \equiv \mathcal{D}'$. Let us show that I'_A can be permuted down I'_1 to obtain the derivation \mathcal{D}_1 . The inductive step of the proof is entirely similar. The cases of I'_A being a unary rule, or both I'_A and I'_1 being binary rules, are clear by simple inspection of the rules. If I'_A is a binary rule that introduces $A = A_0\tau A_1$ (or I'_A is a cut over A) and I'_1 is a unary rule introducing a type $C = C_0\pi C_1$ (where τ and π are logical operators) then the relevant fragments of \mathcal{D} and \mathcal{D}' can be represented as follows, where $s\{X_0, \dots, X_i\}$ stands for a hypersequent with distinguished occurrences of the types X_0, \dots, X_i that do not appear necessarily in this order:

$$\begin{array}{ccc}
\mathcal{D} & & \mathcal{D}' \\
\begin{array}{c} \vdots \mathcal{D}_0 \qquad \vdots \mathcal{D}_1 \\ \hline s_0\{A_0\} \quad s_1\{A_1\} \\ \hline s\{A_0\tau A_1\} \end{array} & & \begin{array}{c} \vdots \mathcal{D}'_0 \qquad \vdots \mathcal{D}'_1 \\ \hline s'_0\{A_0\} \quad s'_1\{A_1\} \\ \hline s'\{A_0\tau A_1; C_0; C_1\} \\ \hline s''\{A_0\tau A_1; C_0\pi C_1\} \end{array}
\end{array}$$

The rule I'_A can be permuted below I'_1 if there is $j \in \{0, 1\}$ such that C_0 and C_1 are in (\mathcal{D}'_j) . This is the case because the rule that introduces $C = C_0\pi C_1$ must occur in \mathcal{D} above I_A . Therefore C , C_0 and C_1 belong to $(\mathcal{D}_j) \subseteq eA_j$ for some $j \in \{0, 1\}$ and, since $(\mathcal{D}'_i) \subseteq eA_i$ for $i \in \{0, 1\}$ and $eA_0 \cap eA_1 = \emptyset$, C_0 and C_1 belong both to (\mathcal{D}'_j) . \square

The No Empty Configuration criteria

In this section, we propose a characterization of $\mathbf{HC}_\epsilon^\omega$ proof nets that can be sequentialized in $\mathbf{HC}_{\mathfrak{f}_0}^\omega$ and in $\mathbf{HC}_{\mathfrak{f}}^\omega$, i.e. without making use of empty configurations of sort zero or, respectively, of any sort.

The characterization is similar to the analogous results for proof nets of linear logic (see Definition 1.32 and subsequent discussion), except that we use here trimmings of hyperswitchings and we make reference, in the case of $\mathbf{HC}_{\mathfrak{f}_0}^\omega$, to the sort of the conclusion of the \wp node at which we trim the proof net.

Lemma 4.55 *Let Π be an $\mathbf{HC}_\epsilon^\omega$ proof net that has exactly one conclusion D . Then, there is a subnet \mathcal{S} of Π that has exactly one conclusion C where, furthermore, C is the conclusion of a \wp link. Moreover, if D has sort 0, then \mathcal{S} can be chosen so that its conclusion C has sort 0.*

Proof. Apply the same reasoning used in Lemma 1.43, ignoring reference to unary operators. Observe that if D has sort 0 and Π is in splitting conditions, then the subnet \mathcal{S} can be chosen to have conclusion of sort 0 because of the restrictions of the sort on the premises of the link. \square

Theorem 4.56 *Let Π be an $\mathbf{HC}_\epsilon^\omega$ proof net. Then, the following facts are equivalent:*

1. for every \wp link L there is a trimming τ at L that contains some conclusion of Π ;
2. all subnets of Π have at least an input conclusion;
3. Π admits a sequentialization in $\mathbf{HC}_{\mathfrak{f}}^\omega$;
4. all sequentializations of Π are in $\mathbf{HC}_{\mathfrak{f}}^\omega$.

Proof. Observe that property 4 immediately entails 3 and that 4 follows from 2. To check all equivalences, it will be enough to prove the following implications: $1 \Rightarrow 2 \Rightarrow 3 \Rightarrow 1$.

$1 \Rightarrow 2$: Suppose, by contraposition, that there is a subnet \mathcal{S} of Π that has only one conclusion. Because of Lemma 4.55 we may assume without loss of generality that the conclusion of \mathcal{S} is the conclusion of a \wp link L . Observe

that no parameter cycle supported by L can go through L 's conclusion, for otherwise there would be a cyclic hyperswitching. Therefore no trimming at L contains conclusions of Π .

$2 \Rightarrow 3$: suppose, by contraposition, that the $\mathbf{HC}_\epsilon^\omega$ proof net Π cannot be sequentialized in $\mathbf{HC}_\epsilon^\omega$, i.e. assume that any sequentialization of Π makes use of the empty configuration Λ_n of sort n . Choose a sequentialization \mathcal{D} and let $\Lambda_n \Rightarrow A$ be a sequent proved by a subderivation \mathcal{E} of \mathcal{D} . Then (\mathcal{E}) is a subnet of Π with no input conclusion.

$3 \Rightarrow 1$: The proof is by induction over the complexity n of Π , defined as the number of logical and cut links in Π . For $n = 0$ –and more in general when there are no \wp links– there is nothing to show. Consider the case that $n \geq 1$ and there are some \wp links. Suppose that in Π there is a final \wp link M . Consider a \wp link L of Π . If $L = M$, any trimming at L contains some conclusion of Π because Π contains some conclusion other than L 's conclusion. Suppose $L \neq M$. Removing M yields an $\mathbf{HC}_\epsilon^\omega$ Π' that can still be sequentialized in $\mathbf{HC}_\epsilon^\omega$. By induction hypothesis there is in Π' a trimming τ' at L that contains some conclusion of Π' . Clearly τ' can be extended to a trimming τ of Π that contains some of its conclusions. Suppose now that Π is in splitting conditions. Then it splits into two subnets both of which can be sequentialized in $\mathbf{HC}_\epsilon^\omega$. One of the subnets contains a \wp link L and, by induction hypothesis, there must be in the subnet a trimming τ' at L that contains some conclusion of the subnet. Clearly, it is possible to extend the (switching underlying the) trimming so as to obtain a trimming τ at L that contains a conclusion of Π . \square

The previous characterization can be adapted straightforwardly to the case of $\mathbf{HC}_{\epsilon_0}^\omega$. To state the result in a concise way, let us introduce two new terms.

Definition 4.57 *A 0-link in an $\mathbf{HC}_\epsilon^\omega$ proof structure is a link the conclusion of which has sort zero. A 0-subnet of an $\mathbf{HC}_\epsilon^\omega$ proof structure is a subnet, the output conclusion of which has sort zero.*

Theorem 4.58 *Let Π be an $\mathbf{HC}_\epsilon^\omega$ proof net. Then, the following facts are equivalent:*

1. *for every \wp 0-link L , there is a trimming τ at L that contains some conclusion of Π ;*
2. *all 0-subnets of Π have at least one input conclusion;*

3. Π admits a sequentialization in $\mathbf{HC}_{\xi_0}^\omega$;
4. all sequentializations of Π are in $\mathbf{HC}_{\xi_0}^\omega$.

Proof. Apply the same reasoning of the previous theorem, making reference to configurations of sort zero (rather than any n) and to $\mathbf{HC}_{\xi_0}^\omega$ instead of \mathbf{HC}_ξ^ω . When proving the implication $1 \Rightarrow 2$, observe that the conclusion of the subnet \mathcal{S} can be assumed, without loss of generality, too have sort zero because of Lemma 4.55. \square

Chapter 5

Bracketed Pregroups

Pregroup grammars are based on partially ordered sets endowed with an algebraic structure ([62, and references therein]). Given a lexicon L , a list of lexical items constitutes therein a sentence of type s if the product, in the given order, of the elements of the pregroup associated in L to the lexical items is less or equal to s . Thus the partial order underlies a computation reminiscent of the Chomskyan notion of derivation ([24]).¹

The roots of this approach, however, lie in logics. Indeed, pregroups are models for Compact Bilinear Logic [59], a logic obtained collapsing the *par* and *tensor* operators of Abrusci's Noncommutative Linear Logic ([1]). As a consequence of the identification, proof nets for this logic are simply planar asymmetric identity links, where the first conclusion of each link is the right-negation of its second conclusion.

The translation of Lambek types into pregroups is non-conservative, as remarked in [18, 16] where the following example is provided: $(p/((p/p)/p))/p \Rightarrow p$ for p atomic cannot be proved in the Lambek Calculus (with or without the empty sequent) but its translation $pp^llp^llp^lp^l \leq p$ holds in any pregroup.

Grammars based on the calculus of free pregroups are weakly equivalent to context-free grammars ([16]) and thus to the Lambek Calculus. Stabler has proposed a generalized system, a particular kind of *Tupled Pregroup Grammars* [116], that is mildly context sensitive, in Joshi's sense. Moortgat and Oehrle [72] have proposed a way to introduce multimodality in pregroup grammars.

¹We are speaking here of a *moral* equivalent. For a reconstruction of the Minimalist Program in terms of sublinear logic see [117, 64] and for a discussion of the relation between the type-logical approach and the minimalist approach see [49].

But in [63] Lambek, responding to the examples cited in the latter article with alternative analyses, has argued that there is no convincing empirical evidence for the introduction of multimodalities, basing his preference for the original proposal on its formal simplicity and on the descriptive results so far obtained for a variety of languages (see, e.g., [8, 7, 9]).²

To solve problems related with associativity Lambek proposes the use of *punctuation symbols*, that have the effect of blocking a derivation ([62]). To formalize this notion we propose here a generalization of pregroups, adapting from multimodal Lambek Calculi the notion of unary modalities, called here *brackets* and *antibrackets*. Marking an element of a pregroup with a bracket has the effect, so to speak, of moving it to a different level: contractions – cancellation of an element by its negation- are available inside brackets, but not across them. This is similar to the use of a depth parameter proposed in [72], but it is achieved inside the language of a poset endowed with an algebraic structure.

Moreover, the bracket pregroup approach is compatible with Francez and Kaminski’s proposal of *commutation-augmented* pregroups. Indeed their theory –glossing over (important!) technical details– is obtained imposing, on freely generated pregroups, constraints that allow a limited amount of commutativity and cancelability ([38]). In this way they obtain pregroups on which they can define grammars that are mildly context sensitive. The very same equations can be imposed unproblematically onto the freely generated pregroups with brackets and antibrackets. The theory of proof nets of bracketed pregroups, however, is no longer valid since the commuting inequations yield crossing identity links.

This chapter is based on my publication [35], adding some comments and improvements that allow for a comparison with Buszkowski’s formalization of Compact Bilinear Logic ([17]). On the one hand, I have simplified the inductive clause of the definition of the free β -pregroup generated by a poset, following the work of Kiślak-Malinowska ([51]). On the other hand, I have expanded the section on Compact Bilinear Logic, including the extension \mathbf{CBL}_{\leq} . This is obtained considering a partially ordered set (P, \leq_P) of atomic symbols and replacing the set of identity axioms $A \Rightarrow A$ with the larger set of axioms of the form $A \Rightarrow B$ for any $A \leq_P B$.

It thus becomes possible to prove, for any $A \leq_P B$, the validity of the following countably many inference schemas (the integer in brackets is a form of negation, as explained later):

²Notice also that Lambek does not believe in a truth-conditional approach to semantics and thus he is not bothered by the lack of the Curry-Howard homomorphism for his new proposal.

$$\frac{\Gamma_1 B^{(2n)} \Gamma_2 \Rightarrow \Delta}{\Gamma_1 A^{(2n)} \Gamma_2 \Rightarrow \Delta} \qquad \frac{\Gamma \Rightarrow \Delta_1 A^{(2n)} \Delta_2}{\Gamma \Rightarrow \Delta_1 B^{(2n)} \Delta_2}$$

$$\frac{\Gamma_1 A^{(2n+1)} \Gamma_2 \Rightarrow \Delta}{\Gamma_1 B^{(2n+1)} \Gamma_2 \Rightarrow \Delta} \qquad \frac{\Gamma \Rightarrow \Delta_1 B^{(2n+1)} \Delta_2}{\Gamma \Rightarrow \Delta_1 A^{(2n+1)} \Delta_2}$$

This is, in my opinion, the only significant difference with Buszkowski's formalization, where (the equivalent of) the previous four schemas have to be postulated as rules.³

The chapter has the following structure. The first section covers background information and gives a sequent presentation of Compact Bilinear Logic, hereafter referred to as **CBL**. In the second section, the notion of β -pregroups is introduced and linguistically motivated. The construction of the free β -pregroup is explained and proof nets, a geometrical method for computing simplifications of pregroup expressions, are defined. The third section shows that the generalization from pregroups to β -pregroups is natural, in the sense that it arises from the algebraization of an extension of compact bilinear logic, for which the cut-elimination theorem is proved.

5.1 Compact bilinear logic and pregroups

The first part of this section is a brief review of the connection between the Lambek Calculus [57], a fragment of Non-commutative Multiplicative-Additive Linear Logic [1], and an algebraic structure known as pregroup [61]. In the following parts, we propose an axiomatisation of Compact Bilinear Logic, both with a two- and a one-sided presentation, and we present a proof of the Cut-Elimination Theorem for Compact Bilinear Logic.⁴

5.1.1 From the Lambek Calculus to Pregroups

Much research in categorial grammar is based on a logic known as the *Lambek Calculus* [57], that we recall here in its Gentzen style presentation.

³To be more precise, the inference rules (that correspond to our identity axioms) and these rules are subsumed by more general inference rules, of which there are countably many for each $A \leq_P B$.

⁴Buszkowski reports in [17] a proof of the Cut Elimination Theorem that he had presented at LACL 2001 in Le Croisic. I prove the theorem with my axiomatisation to pave the way for the generalization of the result for Compact β -Bilinear Logic, see Section 5.3.

Definition 5.1 Let $P = \{A, B, \dots\}$ be a set of propositional symbols. The **LC** types are defined inductively as follows:

$$F := A|F_1/F_2|F_1\backslash F_2|F_1 \times F_2.$$

LC sequents are of the form $\Gamma \Rightarrow F$, where Γ is a finite string of types and F is a type. The rules for the sequent calculus are given in Figure 5.1.

identity and cut	
$\frac{}{A \Rightarrow A} \quad \text{where } A \in P$	$\frac{\Gamma \Rightarrow A \quad \Gamma_1 A \Gamma_2 \Rightarrow C}{\Gamma_1 \Gamma \Gamma_2 \Rightarrow C}$
forward and backward slashes	
$\frac{\Delta \Rightarrow B \quad \Gamma_1 A \Gamma_2 \Rightarrow C}{\Gamma_1 A/B \Delta \Gamma_2 \Rightarrow C}$	$\frac{\Gamma B \Rightarrow A}{\Gamma \Rightarrow A/B}$
$\frac{\Delta \Rightarrow B \quad \Gamma_1 A \Gamma_2 \Rightarrow C}{\Gamma_1 \Delta B \backslash A \Gamma_2 \Rightarrow C}$	$\frac{B \Gamma \Rightarrow A}{\Gamma \Rightarrow B \backslash A}$
product	
$\frac{\Gamma_1 A B \Gamma_2 \Rightarrow C}{\Gamma_1 A \times B \Gamma_2 \Rightarrow C}$	$\frac{\Gamma \Rightarrow A \quad \Delta \Rightarrow B}{\Gamma \Delta \Rightarrow A \times B}$

Figure 5.1: Sequent rules for the Lambek calculus **LC**

The Lambek calculus is a fragment⁵ of the multiplicative part of Non-commutative Multiplicative Additive Linear Logic [1]. We recall here only the multiplicative fragment of the logic.

Definition 5.2 Let P be a set of propositional variables, 0 and 1 two constants. The types of the logic are defined inductively as follows:

$$F := A|0|1|F^l|F^r|F_1 \times F_2|F_1 + F_2$$

where for all types F and G one postulates $F = F^{lr} = F^{rl}$, $(F \times G)^n = G^n + F^n$, and $(F + G)^n = G^n \times F^n$ (where $n = l, r$). Sequent are of the form $\Gamma \Rightarrow \Delta$ where Γ and Δ are not both the null string. The rules for the sequent calculus are given in Figure 5.2.

In non-commutative linear logic one can define directional implications in terms of sum and negation: $A/B := A + B^l$ and $B \backslash A := B^r + A$. The following derivations show that the schemata that correspond to the Lambek

⁵The fragment is conservative if, unlike in the original definition of the Lambek Calculus, the definition of sequents of the Calculus allows for empty strings.

identity	
$\frac{}{A \Rightarrow A}$ where $A \in P$	
cut	
$\frac{\Gamma \Rightarrow \Delta_1 A \Delta_2 \quad \Gamma_1 A \Gamma_2 \Rightarrow \Delta}{\Gamma_1 \Gamma_2 \Rightarrow \Delta_1 \Delta \Delta_2}$ where $\Gamma_1 = \Gamma_2 = \emptyset, \Gamma_1 = \Delta_2 = \emptyset, \Delta_1 = \Gamma_2 = \emptyset,$ or $\Delta_1 = \Delta_2 = \emptyset$	
left- and right-negation	
$\frac{\Psi \Rightarrow F \Delta}{F^l \Psi \Rightarrow \Delta}$	$\frac{\Delta F \Rightarrow \Psi}{\Delta \Rightarrow \Psi F^l}$
$\frac{\Psi \Rightarrow \Delta F}{\Psi F^r \Rightarrow \Delta}$	$\frac{F \Delta \Rightarrow \Psi}{\Delta \Rightarrow F^r \Psi}$
one and zero	
$\frac{}{\Rightarrow 1}$	$\frac{\Gamma_1 \Gamma_2 \Rightarrow \Delta}{\Gamma_1 1 \Gamma_2 \Rightarrow \Delta}$
$\frac{\Gamma \Rightarrow \Delta_1 \Delta_2}{\Gamma \Rightarrow \Delta_1 0 \Delta_2}$	$\frac{}{0 \Rightarrow}$
product and par	
$\frac{\Gamma_1 A B \Gamma_2 \Rightarrow \Delta}{\Gamma_1 A \times B \Gamma_2 \Rightarrow \Delta}$	$\frac{\Gamma_1 \Rightarrow \Delta_1 A \Delta_2 \quad \Gamma_2 \Rightarrow \Delta_3 B \Delta_4}{\Gamma_1 \Gamma_2 \Rightarrow \Delta_3 \Delta_1 A \times B \Delta_4 \Delta_2}$ where either $\Delta_2 = \Delta_3 = \emptyset,$ $\Delta_2 = \Gamma_1 = \emptyset,$ or $\Gamma_2 = \Delta_3 = \emptyset$
$\frac{\Delta_1 A \Delta_2 \Rightarrow \Gamma_1 \quad \Delta_3 B \Delta_4 \Rightarrow \Gamma_2}{\Delta_3 \Delta_1 A + B \Delta_4 \Delta_2 \Rightarrow \Gamma_1 \Gamma_2}$ where either $\Delta_2 = \Delta_3 = \emptyset,$ $\Delta_2 = \Gamma_1 = \emptyset,$ or $\Gamma_2 = \Delta_3 = \emptyset$	$\frac{\Delta \Rightarrow \Gamma_1 A B \Gamma_2}{\Delta \Rightarrow \Gamma_1 A + B \Gamma_2}$
Figure 5.2: Non-commutative Multiplicative Linear Logic	

Calculus axioms for the backward and forward slashes are theorems of Non-commutative Multiplicative Linear Logic:

$$\frac{\Gamma B \Rightarrow A}{\frac{\Gamma \Rightarrow A B^l}{\Gamma \Rightarrow A + B^l}} \qquad \frac{\Delta \Rightarrow B}{\frac{B^l \Delta \Rightarrow \Gamma_1 A \Gamma_2 \Rightarrow C}{\Gamma_1 A + B^l \Gamma_2 \Rightarrow C}}$$

It has been observed ([59]) that pregroups are models for Non-commutative Multiplicative Linear Logic in which \times and 1 have the same interpretation as, respectively, $+$ and 0 .

Definition 5.3 A pregroup is an algebraic structure $(P, \times, 1, l, r; \leq)$ such that:

1. $(P, \times, 1)$ is a monoid, i.e. \times is an associative binary operation in P with unit 1;
2. (P, \leq) is a partially ordered set such that, for any a, b, c, d in P , if $a \leq b$ and $c \leq d$ then $a \times c \leq b \times d$;
3. l and r are unary operations that for any a in P satisfy the equations:

$$a^l \times a \leq 1 \leq a \times a^l, \text{ and } a \times a^r \leq 1 \leq a^r \times a.$$

On the other hand, one could define a logic [59] where there is only one operator that has the property of both the product \times and the sum $+$ and a constant that has the properties of both 0 and 1. In this way, there is virtually no difference between the type $A \times B (= A + B)$, $1 (= 0)$ and, respectively, the string AB and the empty string. This means that we can actually dispense altogether with the operators \times , $+$ and the constants 0 and 1. This logic is the subject of the next subsection. As the reader can verify, its models are pregroups.

5.1.2 Compact Bilinear Logic I

With the identifications suggested above, the rules for sum and multiplication collapse into one rule, that we call here concatenation. Rules for constants are no longer needed. The other rules are left unchanged.

Definition 5.4 (CBL – two-sided presentation) *Let $P = \{A, B, \dots\}$ be a set of propositional variables. The CBL types and the strings of CBL types are defined inductively as follows:*

types: $F := A|F^l|F^r$ where, for all types F , one postulates $F = F^{lr} = F^{rl}$;

strings: $\Gamma := \emptyset|F|\Gamma_1\Gamma_2|\Gamma^l|\Gamma^r$

where the negation Γ^n of a string (n stands for l or r) is defined inductively by $\emptyset^n = \emptyset$ and $(\Gamma G)^n = G^n \Gamma^n$. Sequents are of the form $\Gamma \Rightarrow \Delta$ where $\Gamma \Delta \neq \emptyset$. The rules for the sequent calculus are given in Figure 5.3.

One might be tempted to drop the stipulation $F^{lr} = F^{rl} = F$, since the equivalence of F^{lr} , F^{rl} , and F can be proved. In a similar vein, instead of defining explicitly the negation of a string, one might want to state the negation rules replacing the symbol F for type with the more generic symbol

identity	
$\frac{}{A \Rightarrow A} \quad \text{where } A \in P$	
cut	
$\frac{\Gamma \Rightarrow \Delta_1 F \Delta_2 \quad \Gamma_1 F \Gamma_2 \Rightarrow \Delta}{\Gamma_1 \Gamma_2 \Rightarrow \Delta_1 \Delta_2}$	
where $\Gamma_1 = \Gamma_2 = \emptyset$ or $\Gamma_1 = \Delta_2 = \emptyset$ or $\Delta_1 = \Gamma_2 = \emptyset$ or $\Delta_1 = \Delta_2 = \emptyset$	
left and right negation	
$\frac{\Psi \Rightarrow F \Delta}{F^l \Psi \Rightarrow \Delta}$	$\frac{\Delta F \Rightarrow \Psi}{\Delta \Rightarrow \Psi F^l}$
$\frac{\Psi \Rightarrow \Delta F}{\Psi F^r \Rightarrow \Delta}$	$\frac{F \Delta \Rightarrow \Psi}{\Delta \Rightarrow F^r \Psi}$
concatenation	
$\frac{\Gamma_1 \Rightarrow \Gamma_2 \quad \Delta_1 \Rightarrow \Delta_2}{\Gamma_1 \Delta_1 \Rightarrow \Gamma_2 \Delta_2}$	

Figure 5.3: **CBL**: two-sided presentation

Γ for string, thus defining a logic where one can prove the equivalence of $(\Gamma G)^n$ and $G^n \Gamma^n$ (where $n = l, r$). These, however, would be mistakes, if one wants to define a logic for which the cut-elimination theorem holds. Indeed, without the stipulation about the mixed double negation of a type or, respectively, with the changes concerning the negation of a string, there would be no cut-free proof of, e.g., the following theorems: $A^{lr} A \Rightarrow \emptyset$, and $(AB^l)^r C \Rightarrow B(C^l A)^r$.

Observe that we could enrich **CBL** with an operator \cdot defined by two further rules, the schemas of which correspond, in Non-Commutative Multiplicative Linear Logic, to the rule on the left for \times and on the right for $+$:

$$\frac{\Gamma_1 AB \Gamma_2 \Rightarrow \Delta}{\Gamma_1 A \cdot B \Gamma_2 \Rightarrow \Delta} \qquad \frac{\Delta \Rightarrow \Gamma_1 AB \Gamma_2}{\Delta \Rightarrow \Gamma_1 A \cdot B \Gamma_2}$$

An application of the concatenation rule and the above rules establishes the equivalence $A \cdot B \Leftrightarrow AB$ for any type A and B . Moreover, one can prove the validity of the inference rules that correspond, in Non-Commutative Multiplicative Linear Logic, to the rule on the left for $+$ and on the right for \times . For instance, the case $\Delta_2 = \Delta_3 = 0$ is proved as follows:

$$\frac{\frac{\Delta_1 A \Rightarrow \Gamma_1 \quad B \Delta_4 \Rightarrow \Gamma_2}{\Delta_1 AB \Delta_4 \Rightarrow \Gamma_1 \Gamma_2}}{\Delta_1 A \cdot B \Delta_4 \Rightarrow \Gamma_1 \Gamma_2} \qquad \frac{\frac{\Gamma_1 \Rightarrow \Delta_1 A \quad \Gamma_2 \Rightarrow \Delta_3 B \Delta_4}{\Gamma_1 \Gamma_2 \Rightarrow \Delta_1 AB \Delta_4}}{\Gamma_1 \Gamma_2 \Rightarrow \Delta_1 A \cdot B \Delta_4}$$

Therefore, the operator \cdot has both the properties of the $+$ and the \times operators.

In a similar vein, one could introduce in **CBL** the constant ϵ defined by the following rules, the schemas of which correspond –in Non-Commutative Multiplicative Linear Logic– to the introduction rule of 0 on the left and of 1 on the right.

$$\frac{}{\epsilon \Rightarrow} \qquad \frac{}{\Rightarrow \epsilon}$$

Observe that ϵ is equivalent to the empty sequent and that the other introduction rules for 0 and 1 can be proved as follows:

$$\frac{\frac{\frac{\Gamma_1 \Gamma_2 \Rightarrow \Delta}{\Gamma_2 \Rightarrow \Gamma_1^r \Delta} \quad \frac{}{\epsilon \Rightarrow}}{\epsilon \Gamma_2 \Rightarrow \Gamma_1^r \Delta}}{\Gamma_1 \epsilon \Gamma_2 \Rightarrow \Delta} \qquad \frac{\frac{\frac{\Gamma \Rightarrow \Delta_1 \Delta_2}{\Gamma_1 \Delta_2^r \Rightarrow \Delta_1} \quad \frac{}{\Rightarrow \epsilon}}{\Gamma_1 \Delta_2^r \Rightarrow \Delta_1 \epsilon}}{\Gamma_1 \Rightarrow \Delta_1 \epsilon \Delta_2}$$

Therefore, ϵ has both the properties of 0 and 1.

Enriching **CBL** with the mentioned operators yields a logic, the Tarski-Lindebaum algebra of which is a pregroup in which atomic symbols are not comparable.

If the set P of atomic symbols comes equipped with a (non-trivial) partial order, then we define an extension **CBL** $_{\leq}$ of **CBL**, obtained replacing the set of **CBL** identity axioms by the larger set of axioms $A \Rightarrow B$ for any A and B such that $A \leq B$ in P .

For any $A \leq B$, and any integer n , we can prove the theorems $A^{(2n)} \Rightarrow B^{(2n)}$ and $B^{(2n+1)} \Rightarrow A^{(2n+1)}$ where $C^{(m)}$ stands for C negated m times by r (l) if m is positive (negative) and $C^{(0)} = C$. For instance, let us prove $A^{(2n)} \Rightarrow B^{(2n)}$ with n positive. The proof is by induction on n . Observe that:

$$\frac{\frac{\frac{A^{(2n-2)}}{A^{(2n-2)} B^{(2n-1)}} \Rightarrow \frac{B^{(2n-2)}}{A^{(2n-1)}}}{B^{(2n-1)}} \Rightarrow \frac{B^{(2n)} A^{(2n-1)}}{A^{(2n)}} \Rightarrow \frac{B^{(2n)}}{B^{(2n)}}$$

If $n = 1$, the previous lines are a proof of the basic step. If $n > 1$, they are a proof of the inductive step.

In \mathbf{CBL}_{\leq} there are also new valid inference schemas. In particular, for any $A \leq B$ in P and any integer n one can prove the following rules:

$$\frac{\Gamma_1 B^{(2n)} \Gamma_2 \Rightarrow \Delta}{\Gamma_1 A^{(2n)} \Gamma_2 \Rightarrow \Delta} \qquad \frac{\Gamma \Rightarrow \Delta_1 A^{(2n)} \Delta_2}{\Gamma \Rightarrow \Delta_1 B^{(2n)} \Delta_2}$$

$$\frac{\Gamma_1 A^{(2n+1)} \Gamma_2 \Rightarrow \Delta}{\Gamma_1 B^{(2n+1)} \Gamma_2 \Rightarrow \Delta} \qquad \frac{\Gamma \Rightarrow \Delta_1 B^{(2n+1)} \Delta_2}{\Gamma \Rightarrow \Delta_1 A^{(2n+1)} \Delta_2}$$

The proofs are either via cut or by straightforward inductions over $|n|$. For instance, this is the proof of the bottom left schema, in the case that $n = 0$:

$$\frac{\Gamma A^r \Delta \Rightarrow \Phi}{\Gamma A^r \Rightarrow \Phi \Delta^l}$$

$$\frac{\Gamma \Rightarrow \Phi \Delta^l A}{\Gamma \Rightarrow \Phi \Delta^l B}$$

$$\frac{\Gamma B^r \Rightarrow \Phi \Delta^l}{\Gamma B^r \Delta \Rightarrow \Phi}$$

All the other checks are similar.

To prove the cut-elimination theorem it is expedient to give a one-sided presentation of the logic.

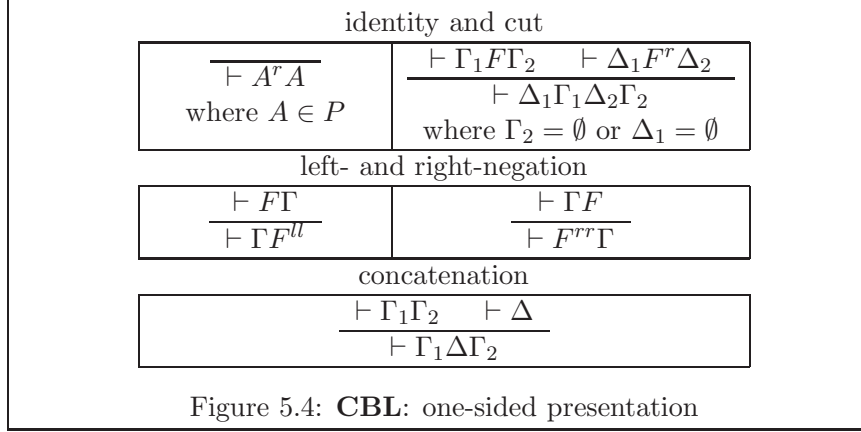
5.1.3 Compact Bilinear Logic II

Definition 5.5 (**CBL** – one-sided presentation) *Let $P = \{A, B, \dots\}$ be a set of propositional variables. The types of the logic and the strings of types are defined inductively as follows:*

types: $F := A | F^l | F^r$ where, for all types F , one postulates $F = F^{lr} = F^{rl}$;

strings: $\Gamma := \emptyset | F | \Gamma_1 \Gamma_2 | \Gamma^l | \Gamma^r$

where the negation $\vdash \Gamma^n$ of a string (n stands for l or r) is defined inductively by $\emptyset^n = \emptyset$ and $(\Gamma G)^n = G^n \Gamma^n$. Sequents are of the form $\vdash \Gamma$ where Γ is not the null string. The rules for the sequent calculus are given in Figure 5.4.



Note that the asymmetry of the axioms with respect to negations is only apparent, since $\vdash AA^l$ can be proved observing that $A^l = A^{rll}$ and using the left negation rule.

As in the previous section, if the set P of propositional variables is partially ordered, we define the one-sided presentation of \mathbf{CBL}_{\leq} substituting the set of **CBL** identity axioms with the larger set of axioms $\vdash A^r B$ for all atomic propositions A and B such that $A \leq B$ in P .

Unsurprisingly, for any $A \leq B$ in P and any integer n , the following theorems and inference schemas hold in \mathbf{CBL}_{\leq} :

$$\begin{array}{cc} \vdash A^{(2n+1)} B^{(2n)} & \vdash B^{(2n+2)} A^{(2n+1)} \\ \\ \frac{\vdash \Gamma A^{(2n)}}{\vdash \Gamma B^{(2n)}} & \frac{\vdash \Gamma B^{(2n+1)}}{\vdash \Gamma A^{(2n+1)}} \end{array}$$

In the following proposition, we show that the two- and the one-sided presentations are equivalent.

Proposition 5.6 *Let Γ and Δ be strings. The sequent $\Gamma \Rightarrow \Delta$ is a theorem of two-sided \mathbf{CBL}_{\leq} if and only if $\vdash \Gamma^r \Delta$ is a theorem of one-sided \mathbf{CBL}_{\leq} . Moreover, $\Gamma \Rightarrow \Delta$ has a cut-free proof if and only if $\vdash \Gamma^r \Delta$ has a cut-free proof.*

Proof: We will prove that:

- (i) if $\Gamma \Rightarrow \Delta$ has a (cut-free) proof of length at most k , then there is a (cut-free) proof of $\vdash \Gamma^r \Delta$;
- (ii) if a sequent $\Gamma^r \Delta$ has a (cut-free) proof of length at most k , then there is a (cut-free) proof of $\Rightarrow \Gamma^r \Delta$ (and thus of $\Gamma \Rightarrow \Delta$).

The proofs are by induction on k . For the base case there is nothing to prove, because if $k = 0$ then $\Gamma \Rightarrow \Delta$ and $\vdash \Gamma^r \Delta$ are instances of the identity axioms. Let us suppose that (i) and (ii) hold for all $n \leq k$.

To prove (i) for $n = k + 1$, let us assume that, in two-sided \mathbf{CBL}_{\leq} , there is a proof of $\Gamma \Rightarrow \Delta$ of length $k + 1$. Let us analyse the last rule ρ used in the proof.

If ρ was an instance of the concatenation rule

$$\frac{\Gamma_1 \Rightarrow \Gamma_2 \quad \Delta_1 \Rightarrow \Delta_2}{\Gamma_1 \Delta_1 \Rightarrow \Gamma_2 \Delta_2}$$

then, by IH (induction hypothesis), there are proofs of $\vdash \Gamma_1^r \Gamma_2$ and of $\vdash \Delta_1^r \Delta_2$ and thus using concatenation one obtains a proof of $\vdash \Delta_1^r \Gamma_1^r \Gamma_2 \Delta_2 = (\Gamma_1 \Delta_1)^r \Gamma_2 \Delta_2$.

If ρ was the negation rule

$$\frac{\Delta F \Rightarrow \Psi}{\Delta \Rightarrow \Psi F^l}$$

then by IH there is a proof of $\vdash F^r \Delta^r \Psi$ and thus of $\vdash \Delta^r \Psi F^l$ (just apply the left negation rule). If the last rule was the introduction rule of a right negation on the left side of the sequent, the proof is similar. For the other negation rules there is nothing to prove.

Note that in all the previous cases, if we start with a cut-free proof we obtain again a cut-free proof. If ρ was an instance of the cut rule

$$\frac{\Gamma \Rightarrow \Delta_1 F \Delta_2 \quad \Gamma_1 F \Gamma_2 \Rightarrow \Delta}{\Gamma_1 \Gamma_2 \Rightarrow \Delta_1 \Delta_2}$$

then, by IH, there are proofs of $\vdash \Gamma^r \Delta_1 F \Delta_2$ and $\vdash \Gamma_2^r F^r \Gamma_1^r \Delta$ and thus, using cut, of $\vdash \Gamma_2^r \Gamma^r \Delta_1 \Gamma_1^r \Delta \Delta_2$. Note that this sequent is indeed equal to $\vdash (\Gamma_1 \Gamma_2)^r \Delta_1 \Delta \Delta_2$, because by the definition of the cut rule either Δ_1 or Γ_1 is the empty string.

Since we have exhausted all the possible case for ρ , the induction is complete for part (i).

The proof of case (ii) is similar. Assume that there is a (cut-free) proof of length $k + 1$ of the sequent $\Gamma^r \Delta$ and analyse the last step ρ in the proof.

If ρ was an instance of the concatenation rule

$$\frac{\vdash \Gamma_1 \Gamma_2 \vdash \Delta}{\vdash \Gamma_1 \Delta \Gamma_2}$$

then there are proofs of $\Rightarrow \Delta$ and $\Gamma_1^l \Rightarrow \Gamma_2$ and thus concatenation yields the proof

$$\frac{\frac{\Rightarrow \Delta \quad \Gamma_1^l \Rightarrow \Delta \Gamma_2}{\Gamma_1^l \Rightarrow \Delta \Gamma_2} \quad \Gamma_1^l \Rightarrow \Gamma_2}{\Rightarrow \Gamma_1 \Delta \Gamma_2}$$

If ρ was an instance of the right negation rule

$$\frac{\vdash \Gamma F}{\vdash F^{rr} \Gamma}$$

then by induction hypothesis there is a proof of $\Rightarrow \Gamma F$ and thus of $\Rightarrow F^{rr} \Gamma$. The check for the case of the left negation rule is similar.

To conclude the proof, all we need to observe is that a one-sided cut

$$\frac{\vdash \Gamma_1 F \Gamma_2 \vdash \Delta_1 F^r \Delta_2}{\vdash \Delta_1 \Gamma_1 \Delta_2 \Gamma_2}$$

can be replaced by the following derivation containing a two-sided cut:

$$\frac{\frac{\Rightarrow \Gamma_1 F \Gamma_2 \quad \frac{\Rightarrow \Delta_1 F^r \Delta_2}{F \Delta_1^l \Rightarrow \Delta_2}}{\Delta_1^l \Rightarrow \Gamma_1 \Delta_2 \Gamma_2}}{\Rightarrow \Delta_1 \Gamma_1 \Delta_2 \Gamma_2}$$

5.1.4 The Cut Elimination Theorem for \mathbf{CBL}_{\leq}

Theorem 5.7 (Cut Elimination Theorem) \mathbf{CBL}_{\leq} enjoys the cut elimination property.

Proof: We will show that every proof α of a sequent $\vdash \Gamma$ of one-sided \mathbf{CBL}_{\leq} can be transformed in a finite number of steps into a cut-free proof β of $\vdash \Gamma$. The analogous result for two-sided \mathbf{CBL}_{\leq} follows then from the previous proposition. Without loss of generality we may assume that the proof α has the following form, where γ and δ are cut-free proofs:

$$\frac{\frac{\gamma}{\vdash \Gamma_1 F \Gamma_2} \quad \frac{\delta}{\vdash \Delta_1 F^r \Delta_2}}{\vdash \Delta_1 \Gamma_1 \Delta_2 \Gamma_2} \text{ cut}$$

We can then prove by induction on the length n of the derivation α that α can be transformed in a finite number of steps into a cut free proof β of $\vdash \Gamma$.

Observe that if γ is the axiom $\vdash A^r B$ for some $A \leq B$ in P , then either $F = A^r$ or $F = B$. Consider the case $F = B$ (the other case is similar). Note that B^r must have been introduced in δ by an axiom $\vdash B^r C$ for some C in P such that $B \leq C$. Hence $A \leq C$ and there is an axiom $\vdash A^r C$. The derivation δ' , obtained replacing in δ the axiom $\vdash B^r C$ with $\vdash A^r C$, is a cut-free proof of $\Delta_1 A^r \Delta_2$.

Thus the base case of the induction, when $n = 1$, is established. Let us suppose that γ has length $n + 1$ and let us analyse the last step ρ in the proof γ of $\vdash \Gamma_1 F \Gamma_2$. Because of the observation of the previous paragraph, we may assume that ρ is not an axiom.

If ρ was an instance of the concatenation rule, then one can switch the order of application of cut and concatenation and obtain a cut-free proof applying the induction hypothesis on the branch of the proof that contains the cut. In more detail, suppose that ρ was an instance of concatenation as in the following schema:

$$\frac{\frac{\zeta}{\vdash \Psi_1 \Psi_2} \quad \frac{\varepsilon}{\vdash \Xi}}{\vdash \Psi_1 \Xi \Psi_2 = \Gamma_1 F \Gamma_2}$$

One has to consider three subcases, depending on whether F appears in Ψ_1 , Ξ , or Ψ_2 .

1.1 If $\Psi_1 = \Psi_{11}F\Psi_{12}$, then $\Gamma_1 = \Psi_{11}$ and $\Gamma_2 = \Psi_{12}\Xi\Psi_2$. Using cut on the last two sequents of ζ and δ and then concatenation, one can write the following proof:

$$\frac{\frac{\varepsilon}{\vdots} \quad \frac{\frac{\zeta}{\vdots} \quad \frac{\delta}{\vdots}}{\vdash \Psi_{11}F\Psi_{12}\Psi_2 \quad \Delta_1 F^r \Delta_2} \text{ cut}}{\vdash \Delta_1 \Psi_{11} \Delta_2 \Psi_{12} \Psi_2} \text{ cut}}{\vdash \Delta_1 \Psi_{11} \Delta_2 \Psi_{12} \Xi \Psi_2 = \Delta_1 \Gamma_1 \Delta_2 \Gamma_2}$$

By induction hypothesis, the boxed derivation can be replaced by a cut-free derivation, thus yielding a cut-free proof of $\Delta_1 \Gamma_1 \Delta_2 \Gamma_2$.

1.2. If $\Psi_2 = \Psi_{21}F\Psi_{22}$, the calculations are similar.

1.3. If $\Xi = \Xi_1F\Xi_2$, the only difference is that one has to apply the cut rule to the last sequents of ε and δ (keeping in mind, to verify the step toward the last line, that either $\Xi_2 = \Psi_2 = \emptyset$ or $\Delta_1 = \emptyset$):

$$\frac{\frac{\zeta}{\vdots} \quad \frac{\frac{\varepsilon}{\vdots} \quad \frac{\delta}{\vdots}}{\vdash \Xi_1 F \Xi_2 \quad \Delta_1 F^r \Delta_2} \text{ cut}}{\vdash \Delta_1 \Xi_1 \Delta_2 \Xi_2} \text{ cut}}{\vdash \Delta_1 \Psi_1 \Xi_1 \Delta_2 \Xi_2 \Psi_2 = \Delta_1 \Gamma_1 \Delta_2 \Gamma_2}$$

If ρ was an instance of the right negation rule, as in the schema below, then there are two subcases.

$$\frac{\frac{\varepsilon}{\vdots}}{\vdash \Gamma G} \quad \frac{\vdash \Gamma G}{\vdash G^{rr} \Gamma = \Gamma_1 F \Gamma_2}$$

2.1. If $\Gamma_1 \neq \emptyset$, then $\Gamma_1 = G^{rr}\Gamma'$ and we can obtain a cut-free proof applying the induction hypothesis to the boxed derivation in the following proof:

$$\frac{\frac{\frac{\varepsilon}{\vdots} \quad \frac{\frac{\delta}{\vdots}}{\Delta_1 F^r \Delta_2}}{F^r \Delta_2 \Delta_1^u} \text{ cut}}{\Gamma' F \Gamma_2 G} \quad \frac{\Gamma' F \Gamma_2 G \quad \frac{\delta}{\vdots}}{\Gamma' \Delta_2 \Delta_1^u \Gamma_2 G} \text{ cut}}{\vdash G^{rr} \Gamma' \Delta_2 \Delta_1^u \Gamma_2} \text{ cut}}{\vdash \Delta_1 \Gamma_1 \Delta_2 \Gamma_2}$$

2.2. If $\Gamma_1 = \emptyset$, then $F = G^{rr}$. We can then apply the same reasoning used in the last subcase, starting from the following proof:

$$\frac{\frac{\frac{\varepsilon}{\vdots} \quad \frac{\frac{\delta}{\vdots}}{\Delta_1 F^r \Delta_2}}{\Delta_2 \Delta_1^u F^L} \text{ cut}}{\Gamma_2 F^{ll}} \quad \frac{\Gamma_2 F^{ll} \quad \frac{\delta}{\vdots}}{\Delta_2 \Delta_1^u \Gamma_2} \text{ cut}}{\vdash \Delta_2 \Delta_1^u \Gamma_2} \text{ cut}}{\vdash \Delta_1 \Delta_2 \Gamma_2}$$

Clearly, if ρ was the left negation rule, the calculations are similar. Thus, having examined all the possibilities, the induction has been established.

5.2 β -pregroup grammars

In this section, we generalize some definitions and results about pregroups [60]. We define the notion of pregroup with β -brackets, explain the free construction of this structure, and provide proof nets. Some examples briefly illustrate the linguistic application of the proposed structure and motivate the further generalization obtained introducing β -antibrackets.

5.2.1 Introducing β -brackets

Multimodal categorial grammars are built either on the non-associative or on the associative Lambek calculus. In the former case a modality can be introduced to license locally associativity. This strategy cannot be followed in the context of pregroups, because, in the absence of associativity, one couldn't even prove modus ponens, since it wouldn't be guaranteed, e.g., that

$(ab^l)b$ simplifies to a . Thus, we will pursue the latter strategy, introducing a method to *break* associativity when needed. The tool we propose, β -brackets and β -antibrackets, is inspired by the notion of bracket semigroup [79].

For expository reasons, we start by presenting the notion of pregroup with β -brackets only.

Definition 5.8 *A pregroup with β -brackets is a pregroup P endowed moreover with a monotone map $\beta : P \rightarrow P$.*

To define grammars based on the notion of pregroup with β -brackets, we need to explain how a partially ordered set (X, \leq_X) freely generates such a structure. Consider the set P' of elements defined inductively as follows (x an element of X , n an integer):

$$p = 1|x^{(n)}|p_1p_2|\beta(p)^{(n)},$$

where 1 stands for the empty string, $p^{(0)}$ for p , and $p^{(n)}$ stands for p negated n times with the left (right) negation, if n is negative (positive).⁶ Consider on P' the following operations:

$$\begin{aligned} \text{multiplication of } p_1 \text{ by } p_2 &= p_1p_2; \\ \beta\text{-bracketing of } p &= \beta(p); \\ \text{left-negation of } p &= p^{(-1)}; \\ \text{right-negation of } p &= p^{(1)} \end{aligned}$$

where for $m = -1, +1$ one defines $1^{(m)} = 1$, $(p^{(n)})^{(m)} = p^{(n+m)}$, and $(p_1p_2)^{(m)} = p_2^{(m)}p_1^{(m)}$. Let $\leq_{P'}$ be the reflexive-transitive closure of the relation \rightarrow defined inductively on P' as follows (where c, d are elements of P'):

$$\begin{aligned} \text{negation contraction: } cu^{(n)}u^{(n+1)}d &\rightarrow cd, \\ \text{negation expansion: } cd &\rightarrow cu^{(n+1)}u^{(n)}d \end{aligned}$$

where $u \in X$ or $u = \beta(u_1)$ for some $u_1 \in P'$, or

$$\begin{aligned} \text{induced step (even case): } cu^{(2n)}d &\rightarrow cv^{(2n)}d, \\ \text{induced step (odd case): } cv^{(2n+1)}d &\rightarrow cu^{(2n+1)}d \end{aligned}$$

⁶We will refer to elements of the form $x^{(n)}$ as simple terms.

where either $u \leq_X v$ or $u = \beta(u_1), v = \beta(v_1)$ and $u_1 \rightarrow v_1$ for some $u_1, v_1 \in P'$.

The relation $\leq_{P'}$ gives rise on P' to an equivalence relation \cong defined as follows: $a \cong b$ if and only if $a \leq_{P'} b$ and $b \leq_{P'} a$. By construction, the equivalence relation is compatible with the operations defined previously on P' . Thus the quotient set $P := P'/\cong$ inherits in a natural way a (partial) order $\leq := \leq_{P'}/\cong$ and all the operations defined in P' . To keep the notations simple, we will refer to an equivalence classe $[a]$ in P directly with the symbol a , and use for the operations in P the same symbols used for the analogous operations in P' .

Note that, without loss of generality, we can always assume that in a chain of contractions, expansions and inductive steps $a = a_0 \leq a_1 \leq \dots \leq a_n = b$ all the contractions precede the expansions. This is because, just as in the case of pregroups [61], if an expansion is followed by a contraction (with, eventually, an induced step intervening in between), either the two rules cancel each other out (leave behind the induced step), or they can be applied in the reverse order. From this observation follows the next result.

Proposition 5.9 *If $a \leq b$ in the pregroup with β -brackets freely generated by a poset X and b is a one-element string, then one can go from a to b by contractions and induced steps only.*

Simplifications in pregroups with β -brackets can be computed adapting from linear logic the method of proof nets. To this purpose, it is convenient to define a set of auxiliary brackets, since this allows for a form of commutativity between negations and brackets.

Definition 5.10 *Let $\beta()$ be a monotone map defined on a pregroup P . For any integer n , we will call n -negated β -bracket the map $\beta^n() : P \rightarrow P$ defined for all a in P by*

$$\beta^n(a) := \beta(a^{(-n)})^{(n)}.$$

An n -negated β -bracket is itself a monotone map. Moreover, it enjoys the following properties:

- for any a in P : $\beta^n(a^{(n)}) = \beta(a)^{(n)}$;
- for any a in P : $\beta^n(a^{(m)})\beta^{n+1}(a^{(m+1)}) \leq 1$.

The following proposition explains when it is possible to simplify n -negated β -brackets.

Proposition 5.11 *Let P be a free pregroup with β -brackets. Then:*

$$\beta^n(a)\beta^m(b) \leq 1 \text{ if and only if } n + 1 = m \text{ and } ab \leq 1$$

Proof: Suppose that $n + 1 = m$ and $ab \leq 1$. Then:

$$\begin{aligned} \text{if } n \text{ is even: } ab \leq 1 &\Rightarrow a^{(-n)} \leq b^{(-n-1)} && \text{(because } a \leq b^1) \\ &\Rightarrow \beta(a^{(-n)}) \leq \beta(b^{(-n-1)}) && (\beta^n(\cdot) \text{ preserves } \leq) \\ &\Rightarrow \beta(a^{(-n)})^{(n)} \leq \beta(b^{(-n-1)})^{(n)} \\ \text{if } n \text{ is odd: } ab \leq 1 &\Rightarrow b^{(-n-1)} \leq a^{(-n)} \\ &\Rightarrow \beta(b^{(-n-1)}) \leq \beta(a^{(-n)}) \\ &\Rightarrow \beta(a^{(-n)})^{(n)} \leq \beta(b^{(-n-1)})^{(n)} \end{aligned}$$

Therefore in both cases $\beta^n(a)\beta^{n+1}(b) = \beta(a^{(-n)})^{(n)} \leq \beta(b^{(-n-1)})^{(n)} \leq 1$.

Suppose that $\beta^n(a)\beta^m(b) \leq 1$. By definition, the expression $\beta^n(a)\beta^m(b)$ is equivalent to $\beta(a^{(-n)})^n\beta(b^{(-m)})^m$. The inequality, therefore, holds exactly when it is possible to apply a generalized contraction, i.e. when $m = n + 1$ and $ab \leq 1$.

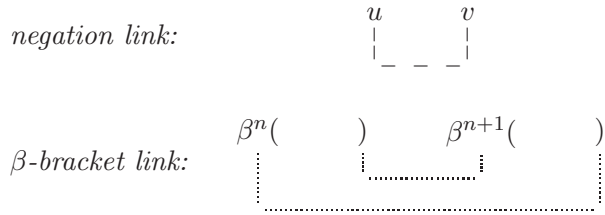
Using n -negated β -brackets, we can define a notion of normal form for the elements of P that will be the base for the definition of proof nets:

$$n := 1|x^{(m)}|n_1n_2|\beta^m(n).$$

The normal form of an element a is obtained going repeatedly through the following steps:

- replace any negation $\beta(x)^{(n)}$ of a β -bracket that appears in a by the n -negated β -bracket $\beta^n(x^{(n)})$;
- replace all negations $(p_1p_2)^{(n)}$ of products by $p_1^{(n)}p_2^{(n)}$, if n is even, and by $p_2^{(n)}p_1^{(n)}$ if n is odd; and,
- replace all negations $(p^{(n)})^{(m)}$ of a negation by the simple term $p^{(n+m)}$.

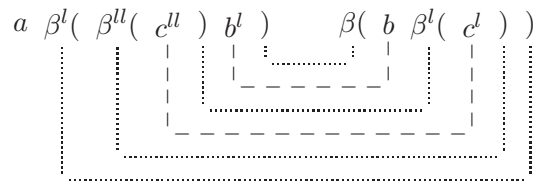
Definition 5.12 (Proof nets) *Let (X, \leq_X) be a poset and P the pregroup with β -brackets freely generated by X . An element a of P , written in normal form, admits of a proof net that reduces to 1 if all simple terms of a and all brackets in a can be connected by planar (i.e. non-intersecting) links of the following form (u and v are simple terms, i.e. negations of elements of X):*



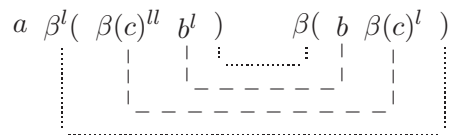
where the above negation link can be placed if there is an element w in X and an integer t such that $u \leq w^{(t)}$ and $v \leq w^{(t+1)}$.

An element a of P , written in normal form, admits of a proof net that reduces to x (x being a one-element string) if there are elements a_1 and a_2 in P such that $a = a_1 x a_2$ and both a_1 and a_2 admit of proof nets that reduce to 1.

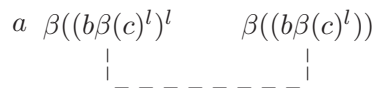
Clearly, it follows from the previous proposition that an element a can be simplified to x if and only if a admits of a proof net that reduces to x . For example, the expression in the next line reduces to a because of the following proof net:



Note that, if we were to accept negation links also on elements that are not basic terms, then we could replace, e.g., three of the above links with just one link, as in the following picture:



And we could even place just one negation link:



Thus, placing bracket links means, so to speak, *looking into the fine structure* of a bracketed expression and creating *channels* that constrain the possible negation links.

In multimodal categorial grammar, data are not simply strings. Rather, they are structured lists. To present the analogous idea in a pregroup context, let us consider a (non-empty) set F of labels (representing the phonological information contained in the lexicon). Let M and M_β be, respectively, the monoid and the monoid with β -brackets freely generated by F . The elements of the former are strings of labels. The elements of the latter are well-bracketed strings of labels. Define by induction the map $q : M_\beta \rightarrow M$ that “forgets brackets”. For all a and b in M_β and ϕ in F let q be defined by $q(\phi) = \phi$, $q(ab) = q(a)q(b)$, and $q(\beta(a)) = q(a)$.

Definition 5.13 (β -structuring of a string) *A β -structuring of a string $\Phi = \phi_1 \dots \phi_n$ of elements of F is an element b of M_β such that $q(b) = \Phi$.*

Note that given a β -structuring b of a string Φ , substituting all occurrences of the ϕ_i s in b with elements τ_i of the free pregroup with β -brackets yields an element of this pregroup. We will denote this element by $b\{\tau_1/\phi_1, \dots, \tau_n/\phi_n\}$.

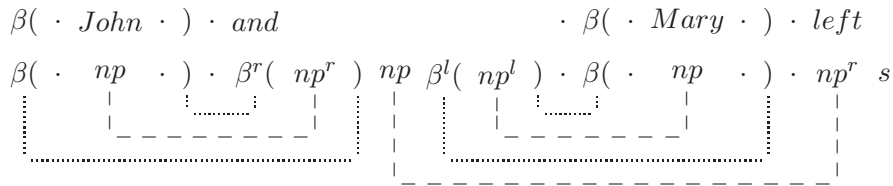
Definition 5.14 (Grammar) *A grammar G based on a pregroup with β -brackets is a quadruple $(F, (X, \leq), T, s)$ where:*

- F is a set (of labels);
- (X, \leq) is a finite partial order;
- T is a map that assigns to each label a finite set of elements of P , where P is the pregroup with β -brackets freely generated by (X, \leq) ; and
- s is an element of X .

A string $\Phi = \phi_1 \dots \phi_n$ of labels is grammatical (according to G) if there is a β -structuring b of F and for all $i = 1, \dots, n$ there are elements $\tau_i \in T(\phi_i)$ such that $b\{\tau_1/\phi_1, \dots, \tau_n/\phi_n\}$ can be simplified to s in P .

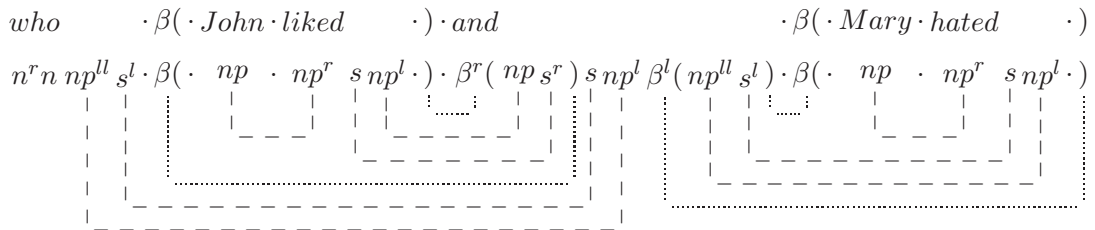
With these definitions in place, we can now illustrate how (negated) β -brackets project islands. Consider for example the type $\beta^r[X^r]X\beta^l[X^l]$ for the conjunction *and*, where X can be instantiated to any type. Then it is

clear that there is only one possibility to structure the following sentence to prove that it is grammatical (according to the proposed typing):⁷



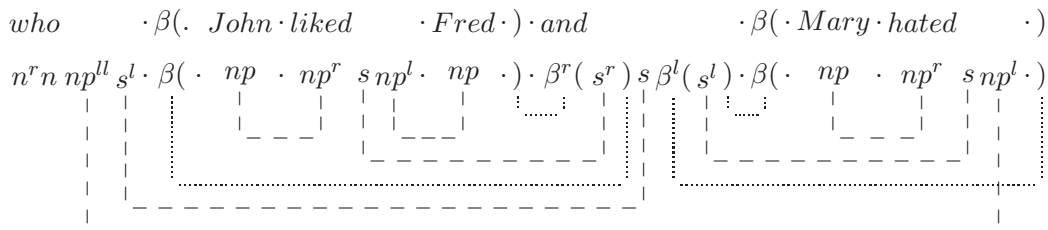
Analogously, one obtains an analysis of the phenomena known as across the board extraction (only the analysis of the relevant fragment of the sentence is presented):

I know the man ...



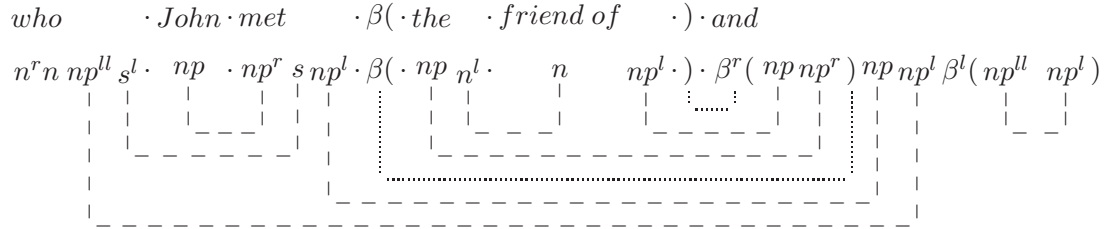
All of this, of course, could have been achieved with a pregroup grammar. The novelty is that the theory proposed here rules out, on syntactic ground, sentences that are not acceptable, because the extraction is not across the board. For instance, if in the following sentence one instantiate X to the symbol s for sentence, then np^{ll} cannot cancel out with any instance of np^l , as reflected by the crossing links in the following picture:

I know the man ...



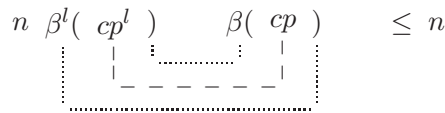
⁷Dots have been inserted in derivations to highlight the correspondence between the lexical items and their types. Brackets that appear also on the first line are due to the β -structuring of the string of labels.

Moreover, the theory proposed here rules out, again on syntactic grounds, the unattested possibility of extracting all the material contained in one conjoined phrase, as in the following example:



The reason for the ungrammaticality in this case is that a bracket $\beta^l(1)$ has not been simplified (not connected with a bracket link).

It should be clear at this point that the structure of a pregroup with β -bracketing allows writing types that force the complements to be bracketed. For example, if a noun selects for a cp-complement, the fact that this complement constitutes an island to extraction can be accounted by the assignment of the type $n\beta^l(cp^l)$ to the noun, thus obtaining the following derivation:



However, the theory is not yet powerful enough to force bracketing of the expression itself together with (some of) its complements. To do this, we need to introduce a notion of antibrackets.

5.2.2 Introducing β -antibrackets

A negated β -bracket cancels out with a β -bracket if the expressions that they contain cancel out as well. The idea behind β -antibracket, by contrast, is that only the brackets cancel out.

Definition 5.15 (β -pregroup) A β -pregroup is a pregroup with β -brackets such that the bracket β has a right adjoint $\hat{\beta}()$, i.e. such that there is a monotone map $\hat{\beta}() : P \rightarrow P$ with the property that, for all a and b in P , $\beta(a) \leq b$ if and only if $a \leq \hat{\beta}(b)$. $\hat{\beta}()$ is called the β -antibracket.

It is immediate to verify that β -antibrackets, if they exist, are uniquely defined and are related to β -brackets by the following rules of expansions and contractions: for all a in P

$$a \leq \widehat{\beta}(\beta(a)) \text{ and } \beta(\widehat{\beta}(a)) \leq a.$$

Given the antibracket map $\widehat{\beta}()$, for any integer n we can define, as has been done for β -brackets, the map $\widehat{\beta}^n : P \rightarrow P$ as follows: for any a in P let $\widehat{\beta}^n(p) = \widehat{\beta}(p^{-n})^{(n)}$. Then it is clear that for all n which are even, $\widehat{\beta}^n()$ is the right adjoint of $\beta^n()$, while for all n which are odd $\beta^n()$ is the right adjoint of $\widehat{\beta}^n()$.

The free β -pregroup generated by a poset (X, \leq) is constructed as the pregroup with β -brackets, except for the following changes:

1. in the recursive definition of the elements of P' there is also a clause for antibrackets:

$$p = 1|x^{(n)}|p_1p_2|\beta(p)^{(n)}|\widehat{\beta}(p)^{(n)};$$

2. on P' one defines also the operation of antibracketing, i.e. the map that to an element p of P' associates the element $\widehat{\beta}(p)$;
3. in defining the relation \rightarrow (and hence $\leq_{P'}$) one has to introduce rules for bracket contraction and expansion and make reference to antibrackets in all clauses (B stands for β or $\widehat{\beta}$):

neg. contraction: $cu^{(n)}u^{(n+1)}d \rightarrow cd$

neg. expansion: $cd \rightarrow cu^{(n+1)}u^{(n)}d$

where $u \in X$ or $u = B(u_1)$ for some $u_1 \in P'$ or

β -contraction: $cu^{(n)}d \rightarrow cv^{(n)}d$

where if n is even $u = \widehat{\beta}(\widehat{\beta}(v))$ and if n is odd $u = \widehat{\beta}(\beta(v))$

β -expansion: $cu^{(n)}d \rightarrow cv^{(n)}d$

where if n is even $v = \widehat{\beta}(\beta(u))$ and if n is odd $v = \beta(\widehat{\beta}(u))$

induced step: $cu^{(n)}d \rightarrow cv^{(n)}d$ where

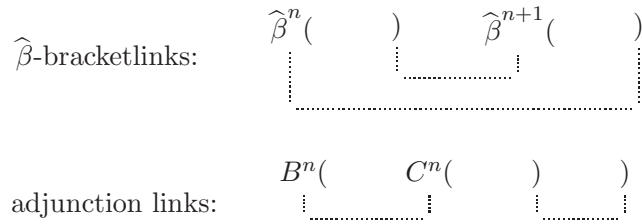
if n is even either $u \leq_X v$ or $u = B(u_1), v = B(v_1)$, and $u_1 \rightarrow v_1$

if n is odd either $v \leq_X u$ or $u = B(u_1), v = B(v_1)$, and $v_1 \rightarrow u_1$

Calculations similar to the previous case show that we can assume without loss of generality that in a sequence of inequalities contractions always precede expansions. Thus we have again the following result.

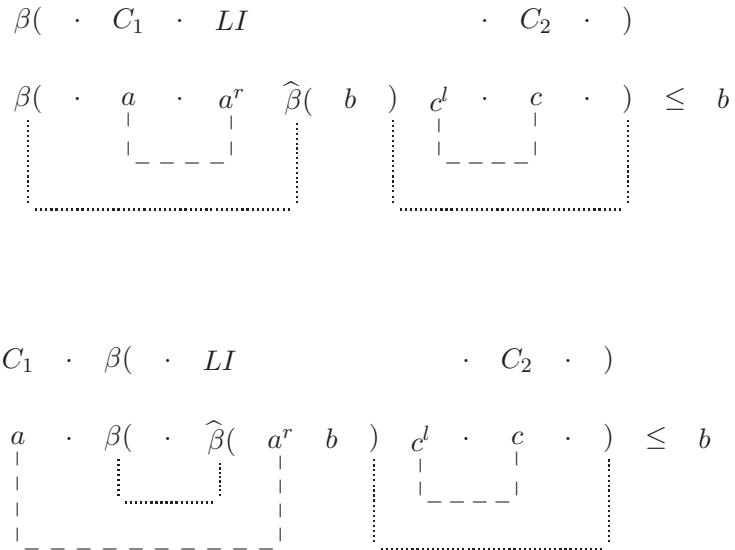
Proposition 5.16 *If $a \leq b$ in the free β -pregroup freely generated by a poset (X, \leq_X) and b is a one-element string, then it is possible to go from a to b by contractions and induced steps only.*

Definition 5.17 (Proof nets for β -pregroups) *The definition reads like the definition for proof nets in pregroup with β -brackets, except that the repertory of channels includes also:*



where, in the last clause, if n is even, then $B = \beta$ and $C = \widehat{\beta}$ and if n is odd, then $B = \widehat{\beta}$ and $C = \beta$.

With this structure, we can now force bracketing also around a lexical item LI and all/some of its complements C_1, C_2, \dots , as pictured below:



5.3 Compact β -bilinear logic

In this section, we propose two extensions of CBL, whose algebraizations give rise respectively to the structure of pregroups with β -brackets and pregroups with β -brackets and antibrackets. For these two logics, we prove the Cut Elimination Theorem.

The structure of pregroup arises from the algebraization of \mathbf{CBL}_{\leq} . This means that it is the result of the following process. Let S be the set of strings of \mathbf{CBL}_{\leq} . It is partially ordered by the relation of implication. Consider the quotient set S/\cong , where \cong is the equivalence relation of coimplication, thus identifying those types that are logically equivalent. Define on S/\cong algebraic operations that correspond to the negation operators of \mathbf{CBL}_{\leq} and to concatenation. For example, the left negation of the set of strings equivalent to a string Γ is defined as the set of strings equivalent to the string Γ^l . The resulting algebra is a pregroup. In a similar vein, one can define logics such that their algebraizations give rise to the notion of pregroup with β -brackets and of pregroup with β -brackets and antibrackets.

We will present here the generalization of the results proved above in the case of compact bilinear logic for β -CBL, the logic corresponding to β -pregroups. The definition and results of the logic corresponding to pregroups with β -brackets only can be derived, dropping all references to $\widehat{\beta}$, from the definition and results given in the following for β -CBL.

We start with the two-sided presentation of Compact β -Bilinear Logic, hereafter referred to as β -CBL.

Definition 5.18 (Two-sided presentation) *Let $P = \{A, B, \dots\}$ be a set of propositional variables. The types of the logic and the strings of types are defined inductively as follows:*

types: $F := A | F^l | F^r | \beta(\Gamma) | \widehat{\beta}(\Gamma)$

where, for all type F , one postulates $F = F^{lr} = F^{rl}$;

strings: $\Gamma := \emptyset | F | \Gamma_1 \Gamma_2 | \Gamma^l | \Gamma^r$

where the negation Γ^n of a string (n stands for l or r) is defined inductively by $\emptyset^n = \emptyset$ and $(\Gamma G)^n = G^n \Gamma^n$. Sequents are of the form $\Gamma \Rightarrow \Delta$ where $\Gamma \Delta \neq \emptyset$. The rules for the sequent calculus are given in Figure 5.5.

identity	
$\frac{}{A \Rightarrow A} \quad \text{where } A \in P$	
cut	
$\frac{\Gamma \Rightarrow \Delta_1 F \Delta_2 \quad \Gamma_1 F \Gamma_2 \Rightarrow \Delta}{\Gamma_1 \Gamma_2 \Rightarrow \Delta_1 \Delta \Delta_2}$ where $\Gamma_1 = \Gamma_2 = \emptyset$ or $\Gamma_1 = \Delta_2 = \emptyset$ or $\Delta_1 = \Gamma_2 = \emptyset$ or $\Delta_1 = \Delta_2 = \emptyset$	
left- and right-negation	
$\frac{\Psi \Rightarrow F \Delta}{F^l \Psi \Rightarrow \Delta}$	$\frac{\Delta F \Rightarrow \Psi}{\Delta \Rightarrow \Psi F^l}$
$\frac{\Psi \Rightarrow \Delta F}{\Psi F^r \Rightarrow \Delta}$	$\frac{F \Delta \Rightarrow \Psi}{\Delta \Rightarrow F^r \Psi}$
concatenation	
$\frac{\Gamma_1 \Rightarrow \Gamma_2 \quad \Delta_1 \Rightarrow \Delta_2}{\Gamma_1 \Delta_1 \Rightarrow \Gamma_2 \Delta_2}$	
bracketing	
$\frac{\Gamma \Rightarrow \Delta}{\beta(\Gamma) \Rightarrow \beta(\Delta)}$	$\frac{\Gamma \Rightarrow \Delta}{\widehat{\beta}(\Gamma) \Rightarrow \widehat{\beta}(\Delta)}$
adjunction	
$\frac{\beta(\Gamma) \Rightarrow \Delta}{\Gamma \Rightarrow \widehat{\beta}(\Delta)}$	$\frac{\Gamma \Rightarrow \widehat{\beta}(\Delta)}{\beta(\Gamma) \Rightarrow \Delta}$
Figure 5.5: Compact β -bilinear logic	

Clearly, β -pregroups are models for β -CBL, since the only new rules are those for bracketing, that guarantee that $\beta()$ and $\widehat{\beta}()$ are compatible with the order, and the adjunction rules, from which it follows that $\widehat{\beta}()$ is the right adjunct of $\beta()$.

As for the case of **CBL**, we could replace the set of identity axioms by the larger set of axioms of the form $A \Rightarrow B$ for any $A \leq B$ in P . The definitions and results carry over straightforwardly from the **CBL** case and therefore we do not overload the presentation with these details.

As for compact bilinear logic, we will present now the logic with one-sided sequents, show that the two presentations are equivalent and thus prove the cut-elimination theorem.

Definition 5.19 (Compact β -bilinear logic) *Let $P = \{A, B, \dots\}$ be a set of propositional variables. The types of the logic and the strings of types are defined inductively as follows:*

types: $F := A|F^l|F^r|\beta(\Gamma)|\widehat{\beta}(\Gamma)$ where, for all F , one postulates $F = F^{lr} = F^{rl}$;

strings: $\Gamma := \emptyset|F|\Gamma_1\Gamma_2|\Gamma^l|\Gamma^r$

where the negation $\vdash \Gamma^n$ of a string (n stands for l or r) is defined inductively by $\emptyset^n = \emptyset$ and $(\Gamma G)^n = G^n \Gamma^n$. Sequents are of the form $\vdash \Gamma$ where Γ is not the null string. The rules for the sequent calculus are given in Figure 5.6.

identity and cut	
$\frac{}{\vdash A^r A}$ where $A \in P$	$\frac{\vdash \Gamma_1 F \Gamma_2 \quad \vdash \Delta_1 F^r \Delta_2}{\vdash \Delta_1 \Gamma_1 \Delta_2 \Gamma_2}$ where $\Gamma_2 = \emptyset$, or $\Delta_1 = \emptyset$
left- and right-negation	
$\frac{\vdash F \Gamma}{\vdash \Gamma F^{ll}}$	$\frac{\vdash \Gamma F}{\vdash F^{rr} \Gamma}$
concatenation	
$\frac{\vdash \Gamma_1 \Gamma_2 \quad \vdash \Delta}{\vdash \Gamma_1 \Delta \Gamma_2}$	
bracketing	
$\frac{\vdash \Gamma^r \Delta}{\vdash \beta(\Gamma)^r \beta(\Delta)}$	$\frac{\vdash \Gamma^r \Delta}{\vdash \widehat{\beta}(\Gamma)^r \widehat{\beta}(\Delta)}$
adjunction	
$\frac{\vdash \beta(\Gamma)^r \Delta}{\vdash \Gamma^r \widehat{\beta}(\Delta)}$	$\frac{\vdash \Gamma^r \widehat{\beta}(\Delta)}{\vdash \beta(\Gamma)^r \Delta}$
Figure 5.6: Compact β -bilinear logic	

The following proposition states the equivalence of the two presentations. Its proof is a straightforward adaptation of the similar result for CBL, and therefore we will not bore the reader with it.

Proposition 5.20 *For any strings Γ and Δ , $\Gamma \Rightarrow \Delta$ is a theorem of two-sided β -CBL if and only if $\vdash \Gamma^r \Delta$ is a theorem of one-sided β -CBL. Moreover, $\Gamma \Rightarrow \Delta$ has a cut-free proof if and only if $\vdash \Gamma^r \Delta$ has a cut-free proof.*

Theorem 5.21 *Compact β -Bilinear Logic enjoys the cut elimination property.*

Proof: As was the case for CBL, all we need to prove is that a cut can be removed from a proof α of one-sided β -CBL that has the following form:

$$\frac{\frac{\gamma}{\vdots} \quad \frac{\delta}{\vdots}}{\frac{\vdash \Gamma_1 F \Gamma_2 \quad \vdash \Delta_1 F^r \Delta_2}{\vdash \Delta_1 \Gamma_1 \Delta_2 \Gamma_2} \text{ cut}}$$

where both subproofs γ and δ are cut-free. The proof is again by induction on the length n of α and all we need to prove are the cases corresponding to the new rules.

If the last rule ρ used in γ was an instance of β -bracketing, then γ has the following form:

$$\frac{\frac{\varepsilon}{\vdots}}{\vdash \Gamma^r \Delta} \quad \frac{\vdash \Gamma^r \Delta}{\vdash \beta(\Gamma)^r \beta(\Delta) = \Gamma_1 F \Gamma_2}$$

There are two subcases to be considered.

1.1. If $F = \beta(\Delta)$ then $\Gamma_1 = \beta(\Gamma)^r$ and $\Gamma_2 = \emptyset$. Therefore we can write the following derivation:

$$\frac{\boxed{\frac{\varepsilon \quad \frac{\delta}{\vdots}}{\frac{\vdash \Gamma^r \Delta \quad \frac{\vdash \Delta_1 \beta(\Delta)^r \Delta_2}{\vdash \beta(\Delta)^r \Delta_2 \Delta_1^u}}{\vdash \Delta^r \widehat{\beta}(\Delta_2 \Delta_1^u)} \text{ cut}}}{\vdash \Gamma^r \widehat{\beta}(\Delta_2 \Delta_1^u)}}{\frac{\vdash \beta(\Gamma)^r \Delta_2 \Delta_1^u}{\vdash \Delta_1 \beta(\Gamma)^r \Delta_2 = \Delta_1 \Gamma_1 \Delta_2}}$$

By induction hypothesis, the boxed subproof above can be substituted by a cut-free proof, thus yielding a cut-free derivation of $\Delta_1 \Gamma_1 \Delta_2$.

1.2. If $F = \beta(\Gamma)^r$ then $\Gamma_1 = \Delta_1 = \emptyset$ and $\Gamma_2 = \beta(\Delta)$. Without loss of generality, we may assume that δ has the following form for some string Ψ , where along the branch of the derivation labelled by η the rules apply to $\beta(\Gamma)$ as one block:

$$\frac{\frac{\frac{\varepsilon}{\vdots}}{\vdash \Gamma^r \Delta}}{\vdash \beta(\Gamma)^r \beta(\Delta) = F\Gamma_2} \quad \boxed{\frac{\frac{\frac{\zeta}{\vdots}}{\vdash \Psi^r \Gamma}}{\vdash \beta(\Psi)^r \beta(\Gamma)}}{\vdash \beta(\Gamma)^{rr} \Delta_2 = F^r \Delta_2}}{\vdash \Delta_2 \Gamma_2}$$

Joining ε and ζ with cut yields a derivation of $\vdash \Psi^r \Delta$. By induction hypothesis, there is a cut-free proof ϑ of this sequent. Consider the boxed subproof pictured above. Replacing ζ with ϑ and any instance of Γ with an instance of Δ , one obtains a cut-free proof of $\beta(\Delta)^{rr} \Delta_2 = \Gamma_2^{rr} \Delta_2$ and therefore of $\Delta_2 \Gamma_2$.

If the last rule ρ used in γ was an instance of the right adjunction rule, then γ has the following form:

$$\frac{\frac{\frac{\varepsilon}{\vdots}}{\vdash \Gamma^r \widehat{\beta}(\Gamma_2)}}{\beta(\Gamma)^r \Delta = \Gamma_1 F \Gamma_2}$$

There are three subcases to be considered.

2.1. If $F = \beta(\Gamma)^r$ then $\Gamma_1 = \emptyset$ and $\Gamma_2 = \Delta$, we can use the same technique as in the previous case:

$$\frac{\frac{\frac{\varepsilon}{\vdots}}{\vdash \Gamma^r \widehat{\beta}(\Gamma_2)}}{\vdash \beta(\Gamma)^r \Gamma_2 = F\Gamma_2} \quad \boxed{\frac{\frac{\frac{\zeta}{\vdots}}{\vdash \Psi^r \Gamma}}{\vdash \beta(\Psi)^r \beta(\Gamma)}}{\vdash \Delta_1 \beta(\Gamma)^{rr} \Delta_2 = \Delta_1 F^r \Delta_2}}{\vdash \Delta_1 \Delta_2 \Gamma_2}$$

The only differences are that now the induction hypothesis will give us a cut-free proof ϑ of $\vdash \Psi^r \widehat{\beta}(\Gamma_2)$ and the substitutions will result in the sequent $\vdash \Delta_1 \Gamma_2^{rr} \Delta_2$, from which $\vdash \Delta_1 \Delta_2 \Gamma_2$ follows, because either Γ_2 or Δ_2 is the empty string.

If F is a factor of the string Δ , then $\Gamma_1 = \beta(\Gamma)^r \Gamma'$ for some string Γ' and $\Gamma' G \Gamma_2 = \Delta$. Hence the sequent before the last in γ is $\vdash \Gamma^r \widehat{\beta}(\Gamma' F \Gamma_2)$. Depending on how the $\widehat{\beta}$ -bracket was introduced, one has to distinguish two cases:

2.2. if $\widehat{\beta}()$ was introduced using a $\widehat{\beta}$ -bracketing rule, then we may assume without loss of generality that α has the following form, where along η the rules apply to $\widehat{\beta}(\Gamma' F \Gamma_2)$ as a whole block:

$$\begin{array}{c}
 \boxed{\begin{array}{c}
 \varepsilon \\
 \vdots \\
 \hline
 \vdash \Psi^r \Gamma' F \Gamma_2 \\
 \hline
 \vdash \widehat{\beta}(\Psi)^r \widehat{\beta}(\Gamma' F \Gamma_2) \\
 \vdots \qquad \qquad \qquad \vdots \\
 \hline
 \vdash \Gamma^r \widehat{\beta}(\Gamma' F \Gamma_2) \\
 \hline
 \vdash \beta(\Gamma)^r \Gamma' F \Gamma_2
 \end{array}}
 \qquad
 \begin{array}{c}
 \delta \\
 \vdots \\
 \hline
 \Delta_1 F^r \Delta_2
 \end{array} \\
 \hline
 \vdash \Delta_1 \Gamma_1 \Delta_2 \Gamma_2
 \end{array}$$

The subproofs ε and δ can be joined with cut to yield a proof of $\vdash \Delta_1 \Psi^r \Gamma' \Delta_2 \Gamma_2$. By induction hypothesis, there is a cut-free proof of this sequent, that moreover can be continued applying a negation rule to yield a proof ϑ of $\vdash \Psi^r \Gamma' \Delta_2 \Gamma_2 \Delta_1^u$. Consider the boxed subproof pictured above. Replacing ε with ϑ and any occurrence of $\Gamma' F \Gamma_2$ with an occurrence of $\Gamma' \Delta_2 \Gamma_2 \Delta_1^u$ one obtains a cut-free proof of $\vdash \Delta_1 \beta(\Gamma)^r \Gamma' \Delta_2 \Gamma_2$, i.e. of $\vdash \Delta_1 \Gamma_1 \Delta_2 \Gamma_2$.

2.3. If $\widehat{\beta}()$ was introduced using an adjunction rule, then we may assume that α has the following form:

$$\begin{array}{c}
 \boxed{\begin{array}{c}
 \varepsilon \\
 \vdots \\
 \hline
 \vdash \beta(\Psi)^r \Gamma' F \Gamma_2 \\
 \hline
 \vdash \Psi^r \widehat{\beta}(\Gamma' F \Gamma_2) \\
 \vdots \qquad \qquad \qquad \vdots \\
 \hline
 \vdash \Gamma^r \widehat{\beta}(\Gamma' F \Gamma_2) \\
 \hline
 \vdash \beta(\Gamma)^r \Gamma' F \Gamma_2
 \end{array}}
 \qquad
 \begin{array}{c}
 \delta \\
 \vdots \\
 \hline
 \Delta_1 F^r \Delta_2
 \end{array} \\
 \hline
 \vdash \Delta_1 \Gamma_1 \Delta_2 \Gamma_2
 \end{array}$$

Then, as in the previous case, there is by induction hypothesis a cut-free proof of the sequent $\vdash \Delta_1 \beta(\Psi)^r \Gamma' \Delta_2 \Gamma_2$ and thus, after making the obvious substitutions, of $\vdash \Delta_1 \Gamma_1 \Delta_2 \Gamma_2$.

The proof of the other cases, needed to establish the induction, are variations on the above calculations and are left to the reader.

Appendix A

Symmetrical Calculi

In this section we review **LGC**, the Lambek-Grishin Calculus as proposed by M. Moortgat on the basis of previous work by V.N. Grishin ([44]). We deal only with the fragment in which the interaction principles (G1) to (G4) hold, i.e. $\mathbf{LG}_\emptyset + \mathcal{G}^\dagger$ according to the notations used in [70]. Following [43], we review the relevant work on displaying the Lambek Calculus. Finally we propose for these displayed calculi a theory of proof nets, where edges are labeled by types and polarities are encoded as directions on the edges. Essentially, this proposal is just a recasting of [77] substituting the notion of tree by the notion of *polar seaweed*.

This proposal, elaborated independently, has been now superseded by R. Moot's work ([75]) and therefore it is not presented in the main body of the thesis, but in this appendix.

A.1 Lambek-Grishin Calculus LGC

Logical operators come in pairs of dual elements $\{\times, \oplus\}$, $\{/ , \oslash\}$, and $\{\backslash, \otimes\}$.

Definition A.1 *The set $\mathcal{T}_{\mathbf{LGC}}(V)$ of **LGC** types is the smallest set that contains a set V of atomic types and that is closed under the binary operations \times , $/$, \backslash , \oplus , \oslash , and \otimes , i.e. the elements of $\mathcal{T}_{\mathbf{LGC}}(V)$ are defined inductively in the following way:¹*

$$\mathcal{T} = V \mid (\mathcal{T} \times \mathcal{T}) \mid (\mathcal{T} / \mathcal{T}) \mid (\mathcal{T} \backslash \mathcal{T}) \mid (\mathcal{T} \oplus \mathcal{T}) \mid (\mathcal{T} \oslash \mathcal{T}) \mid (\mathcal{T} \otimes \mathcal{T}).$$

¹As usual, outer parenthesis will be dropped.

The duality extends to types, according to the following inductive definition:

- if A is atomic, then $A_d = A$; otherwise,
- $(A \diamond B)_d = A_d \diamond_d B_d$ where $\{\diamond, \diamond_d\}$ is any pair of dual operators.

Definition A.2 An **LGC** configuration is an abstract \mathcal{NL} -tree with nodes labeled by the binary operator \circ and leaves labeled by **LGC** types, i.e. **LGC** configurations are defined inductively in the following way:²

$$\tau = \mathcal{T} \mid (\tau \circ \tau).$$

Definition A.3 An **LGC** sequent $\sigma \Rightarrow \tau$ is an ordered pair (σ, τ) of **LGC** configurations.

The dual operator is a homomorphism of **LGC** configurations, i.e. the dual configuration τ_d of a configuration τ is defined by induction as follows:

- if τ is a type \mathcal{T} , then $\tau_d = \mathcal{T}_d$;
- if $\tau = \sigma \circ \rho$, then $\tau_d = \sigma_d \circ \rho_d$.

The dual of a **LGC** sequent $\sigma \Rightarrow \tau$ is defined as $\tau_d \Rightarrow \sigma_d$.

Let H be a symbol that does not belong to \mathcal{L} .

Definition A.4 An **LGC** context is an abstract \mathcal{NL} -tree context $\sigma[H]$ with nodes labeled by \circ , one leaf labeled by H , and the other leaves, if there are any, labeled by **LGC** types.

The result of substituting a configuration ρ in the context $\sigma[H]$ is the configuration $\sigma[\rho]$ defined as follows:

- if $\sigma[H] = H$, then $\sigma[\rho] = \rho$;
- if $\sigma[H] = \xi[H] \circ \tau$, then $\sigma[\rho] = \xi[\rho] \circ \tau$;
- if $\sigma[H] = \tau \circ \xi[H]$, then $\sigma[\rho] = \tau \circ \xi[\rho]$.

identity and cut rules		
Id	$\overline{A \Rightarrow A}$ where A atomic	
cut	$\frac{\rho \Rightarrow A \quad \sigma[A] \Rightarrow \tau}{\sigma[\rho] \Rightarrow \tau}$	$\frac{\rho \Rightarrow \sigma[A] \quad A \Rightarrow \tau}{\rho \Rightarrow \sigma[\tau]}$
logical rules		
\times	$\frac{\sigma[A \circ B] \Rightarrow \tau}{\sigma[A \times B] \Rightarrow \tau}$	$\frac{\sigma \Rightarrow A \quad \tau \Rightarrow B}{\sigma \circ \tau \Rightarrow A \times B}$
\oplus	$\frac{A \Rightarrow \sigma \quad B \Rightarrow \tau}{A \oplus B \Rightarrow \sigma \circ \tau}$	$\frac{\sigma \Rightarrow \tau[A \circ B]}{\sigma \Rightarrow \tau[A \oplus B]}$
\setminus	$\frac{\rho \Rightarrow B \quad \sigma[A] \Rightarrow \tau}{\sigma[\rho \circ B \setminus A] \Rightarrow \tau}$	$\frac{B \circ \sigma \Rightarrow \tau[A]}{\sigma \Rightarrow \tau[B \setminus A]}$
\otimes	$\frac{\tau[A] \Rightarrow B \circ \sigma}{\tau[B \otimes A] \Rightarrow \sigma}$	$\frac{B \Rightarrow \rho \quad \tau \Rightarrow \sigma[A]}{\tau \Rightarrow \sigma[\rho \circ B \otimes A]}$
$/$	$\frac{\rho \Rightarrow B \quad \sigma[A] \Rightarrow \tau}{\sigma[A/B \circ \rho] \Rightarrow \tau}$	$\frac{\sigma \circ B \Rightarrow \tau[A]}{\sigma \Rightarrow \tau[A/B]}$
\odot	$\frac{\tau[A] \Rightarrow \sigma \circ B}{\tau[A \odot B] \Rightarrow \sigma}$	$\frac{B \Rightarrow \rho \quad \tau \Rightarrow \sigma[A]}{\tau \Rightarrow \sigma[A \odot B \circ \rho]}$

Figure A.1: **LGC** rules

Definition A.5 *The rules for the **LGC** calculus are listed in Figure A.1.*

Examples of derivations are given in Figure A.2 on page 187. These derivations, and similar ones, establish the following result.

$\frac{\overline{A \Rightarrow A} \quad \overline{B \Rightarrow B} \quad \overline{C \Rightarrow C}}{\overline{B \circ C \Rightarrow B \times C}}$	$\frac{\overline{A \Rightarrow A} \quad \overline{B \Rightarrow B} \quad \overline{C \Rightarrow C}}{\overline{B \oplus C \Rightarrow B \circ C}}$
$\frac{\overline{B \circ C \Rightarrow A \circ A \otimes (B \times C)}}{\overline{(A \otimes B) \circ C \Rightarrow A \otimes (B \times C)}}$	$\frac{\overline{A \circ A \setminus (B \oplus C) \Rightarrow B \circ C}}{\overline{A \setminus (B \oplus C) \Rightarrow (A \setminus B) \circ C}}$
$\frac{\overline{(A \otimes B) \circ C \Rightarrow A \otimes (B \times C)}}{\overline{(A \otimes B) \times C \Rightarrow A \otimes (B \times C)}}$	$\frac{\overline{A \setminus (B \oplus C) \Rightarrow (A \setminus B) \circ C}}{\overline{A \setminus (B \oplus C) \Rightarrow (A \setminus B) \oplus C}}$

Figure A.2: Deriving (G1) and (G1')

Proposition A.6 *The following sequents are **LGC** theorems:*

²As usual, outer parenthesis will be dropped.

$$\begin{array}{ll}
(\mathbf{G1}) & (A \otimes B) \times C \Rightarrow A \otimes (B \times C); & (\mathbf{G1}') & A \setminus (B \oplus C) \Rightarrow (A \setminus B) \oplus C; \\
(\mathbf{G2}) & C \times (A \otimes B) \Rightarrow A \otimes (C \times B); & (\mathbf{G2}') & A \setminus (C \oplus B) \Rightarrow C \oplus (A \setminus B); \\
(\mathbf{G3}) & C \times (B \circlearrowleft A) \Rightarrow (C \times B) \circlearrowleft A; & (\mathbf{G3}') & (C \oplus B) / A \Rightarrow C \oplus (B / A); \\
(\mathbf{G4}) & (B \circlearrowleft A) \times C \Rightarrow (B \times C) \circlearrowleft A; & (\mathbf{G4}') & (B \oplus C) / A \Rightarrow (B / A) \oplus C.
\end{array}$$

It is well known that this presentation of **LGC** does not enjoy the Cut Elimination Property. Consider, for instance, the example given in [73] where the use of Cut proves to be unavoidable. It is based on the types $i := n \setminus s$ of an intransitive verb, $t := i / n$ of a transitive verb, and $T := (i \circlearrowleft i) \otimes t$, a higher order type for a transitive verb. The sequent $n \circ (T \circ n) \Rightarrow s$ can be proved combining the theorems $T \Rightarrow t$ and $n \circ (t \circ n) \Rightarrow s$ by the cut rule but has no cut-free proof.

In Section A.3, we will define the logic **LGC_d**, a conservative extension of **LGC**, that enjoys the Cut Elimination Property (see Corollary A.38). The embedding into **LGC_d** shows also that **LGC** is a conservative extension of **NLC** (see Corollary A.53).

Definition A.7 *For any LGC configuration σ , the left closure $l\sigma$ and the right closure $r\sigma$ are defined by induction in the following way:*

- if $\sigma = A$ is a trivial configuration, then $lA = rA = A$;
- if $\sigma = \tau \circ \rho$, then $l\sigma = l\tau \times l\rho$ and $r\sigma = r\tau \oplus r\rho$.

Lemma A.8 *For any configuration σ of LGC, the sequents $\sigma \Rightarrow l\sigma$ and $r\sigma \Rightarrow \sigma$ are theorems of LGC.*

Proof. The proof is by induction over the structural complexity of σ . If σ is a trivial configuration, there is nothing to show. If σ is a structural product, then apply the induction hypothesis to the immediate subconfigurations and, for the case of the left (right) closure, apply the tensor right rule (plus left rule) and the (inductive clause of the) definition of left (right) closure. \square

Observe that for any configuration $\sigma[\rho]$, a simple proof by induction over the structural complexity of $\sigma[\]$ shows that $l(\sigma[\rho]) = l(\sigma[l(\rho)])$ and $r(\sigma[\rho]) = r(\sigma[r(\rho)])$.

Lemma A.9 *For any LGC theorem $\sigma \Rightarrow \tau$, the sequents $l\sigma \Rightarrow \tau$ and $\sigma \Rightarrow r\tau$ are LGC theorems.*

Proof. The following proof by induction over the structural complexity of σ shows that $l\sigma \Rightarrow \tau$ is an **LGC** theorem. Indeed, if σ is a trivial configuration, there is nothing to prove. If it is not trivial, then σ can be seen as the configuration $\xi[A \circ B]$ for some suitable context $\xi[\]$ and types A and B . Then any derivation of $\sigma \Rightarrow \tau$ can be continued using the tensor left rule, which yields $\xi[A \times B] \Rightarrow \tau$. Since $\xi[A \times B]$ has a lower structural complexity, by induction hypothesis there is a derivation of $l(\xi[A \circ B]) \Rightarrow \tau$, i.e. $l(\xi[A \circ B]) \Rightarrow \tau$. The proof that $\sigma \Rightarrow r\tau$ is a **LGC** theorem is obtained by duality. \square

The previous two lemmas establish the following result.

Proposition A.10 *For any LGC sequent $\sigma \Rightarrow \tau$, $\sigma \Rightarrow \tau$ is an LGC theorem if and only if $l\sigma \Rightarrow r\tau$ is an LGC theorem.*

A.2 Displayed Lambek Calculus φ -LC_d

Let φ be a possibly empty subset of $\{a, c\}$.

Definition A.11 *The set $\mathcal{T}_{\varphi\text{-LC}_d}(V)$ of $\varphi\text{-LC}_d$ types is the smallest set that contains a set V of atomic types and that is closed under the binary operations \times_{φ} , $/_{\varphi}$, \backslash_{φ} , \oplus_{φ} , \otimes_{φ} , and \odot_{φ} , i.e. the elements of $\mathcal{T}_{\varphi\text{-LC}}(V)$ are defined inductively in the following way:³*

$$\mathcal{T} = V \mid (\mathcal{T} \times_{\varphi} \mathcal{T}) \mid (\mathcal{T} /_{\varphi} \mathcal{T}) \mid (\mathcal{T} \backslash_{\varphi} \mathcal{T}) \mid (\mathcal{T} \oplus_{\varphi} \mathcal{T}) \mid (\mathcal{T} \otimes_{\varphi} \mathcal{T}) \mid (\mathcal{T} \odot_{\varphi} \mathcal{T}).$$

Definition A.12 *A $\varphi\text{-LC}_d$ configuration is an abstract \mathcal{NL} tree with nodes labeled by \circ_{φ} , $>_{\varphi}$, and $<_{\varphi}$, and leaves labeled by $\varphi\text{-LC}_d$ types, i.e. $\varphi\text{-LC}_d$ configurations are defined inductively in the following way:⁴*

$$\mathcal{T} = \mathcal{T} \mid (\mathcal{T} \circ_{\varphi} \mathcal{T}) \mid (\mathcal{T} >_{\varphi} \mathcal{T}) \mid (\mathcal{T} <_{\varphi} \mathcal{T}).$$

Definition A.13 *A $\varphi\text{-LC}_d$ sequent $\tau \Rightarrow \sigma$ is an ordered pair (τ, σ) of $\varphi\text{-LC}_d$ configurations.*

Definition A.14 *The rules for the $\varphi\text{-LC}_d$ calculus comprise:*

³As usual, outer parenthesis will be dropped.

⁴As usual, outer parenthesis will be dropped.

- the rules listed in Figure A.3, where the modality index is $n = \varphi$;
- if $\varphi \neq \emptyset$, the structural rules of Figure A.4.

identity and cut rules			
Id	$\frac{}{A \Rightarrow A}$ where A atomic	Cut	$\frac{\tau \Rightarrow A \quad A \Rightarrow \sigma}{\tau \Rightarrow \sigma}$
display rules			
(display-l)	$\frac{\tau >_n \sigma \Rightarrow \rho}{\sigma \Rightarrow \tau \circ_n \rho}$ $\frac{}{\sigma <_n \rho \Rightarrow \tau}$	(display-r)	$\frac{\rho \Rightarrow \tau >_n \sigma}{\tau \circ_n \rho \Rightarrow \sigma}$ $\frac{}{\tau \Rightarrow \sigma <_n \rho}$
logical rules			
\times_n	$\frac{A \circ_n B \Rightarrow \sigma}{A \times_n B \Rightarrow \sigma}$	\times_n	$\frac{\tau \Rightarrow A \quad \sigma \Rightarrow B}{\tau \circ_n \sigma \Rightarrow A \times_n B}$
\oplus_n	$\frac{A \Rightarrow \tau \quad B \Rightarrow \sigma}{A \oplus_n B \Rightarrow \tau \circ_n \sigma}$	\oplus_n	$\frac{\sigma \Rightarrow A \circ_n B}{\sigma \Rightarrow A \oplus_n B}$
\setminus_n	$\frac{A \Rightarrow \tau \quad \sigma \Rightarrow B}{B \setminus_n A \Rightarrow \sigma >_n \tau}$	\setminus_n	$\frac{\tau \Rightarrow B >_n A}{\tau \Rightarrow B \setminus_n A}$
\odot_n	$\frac{B >_n A \Rightarrow \tau}{B \odot_n A \Rightarrow \tau}$	\odot_n	$\frac{\tau \Rightarrow A \quad B \Rightarrow \sigma}{\sigma >_n \tau \Rightarrow B \odot_n A}$
$/_n$	$\frac{A \Rightarrow \tau \quad \sigma \Rightarrow B}{A /_n B \Rightarrow \tau <_n \sigma}$	$/_n$	$\frac{\tau \Rightarrow A <_n B}{\tau \Rightarrow A /_n B}$
\oslash_n	$\frac{A <_n B \Rightarrow \tau}{A \oslash_n B \Rightarrow \tau}$	\oslash_n	$\frac{\tau \Rightarrow A \quad B \Rightarrow \sigma}{\tau <_n \sigma \Rightarrow A \oslash_n B}$

Figure A.3: Minimal set of rules

Proposition A.15 $\varphi\text{-LC}_d$ is a conservative extension of $\varphi\text{-LC}$.

Proof. The key observation is that any $\varphi\text{-LC}$ rule corresponds, via display, to a $\varphi\text{-LC}_d$ rule. Consider, for instance, the $\varphi\text{-LC}$ rule for the introduction on the left of \setminus_φ :

$$\frac{\rho[A] \Rightarrow C \quad \sigma \Rightarrow B}{\rho[\sigma \circ_\varphi B \setminus_\varphi A] \Rightarrow C}$$

Because of the display rules, the premise $\rho[A] \Rightarrow C$ is equivalent, for some suitable configuration τ , to the sequent $A \Rightarrow \tau$. The conclusion of the rule can then be written equivalently as $\sigma \circ_\varphi B \setminus_\varphi A \Rightarrow \tau$, which in turn can be written as $B \setminus_\varphi A \Rightarrow \sigma >_\varphi \tau$. Thus, displaying the active types, the $\varphi\text{-LC}_d$

structural rules			
(ASS_l) if $a \in \varphi$	$\frac{(\tau_1 \circ_\varphi \tau_2) \circ_\varphi \tau_3 \Rightarrow \tau}{\tau_1 \circ_\varphi (\tau_2 \circ_\varphi \tau_3) \Rightarrow \tau}$	(ASS_r) if $a \in \varphi$	$\frac{\sigma \Rightarrow (\tau_1 \circ_\varphi \tau_2) \circ_\varphi \tau_3}{\sigma \Rightarrow \tau_1 \circ_\varphi (\tau_2 \circ_\varphi \tau_3)}$
(COM_l) if $c \in \varphi$	$\frac{\tau_0 \circ_\varphi \tau_1 \Rightarrow \tau}{\tau_1 \circ_\varphi \tau_0 \Rightarrow \tau}$	(COM_r) if $c \in \varphi$	$\frac{\sigma \Rightarrow \tau_0 \circ_\varphi \tau_1}{\sigma \Rightarrow \tau_1 \circ_\varphi \tau_0}$

Figure A.4: Structural rules of $\varphi\text{-LC}_d$

rule is obtained. Similarly, any structural rule in $\varphi\text{-LC}$ corresponds, via display, to a left structural rule of $\varphi\text{-LC}_d$. \square

A.3 Displayed Multimodal Lambek Calculus $\sigma\text{-LC}_d$

The calculus $\sigma\text{-LC}_d$ is a multimodal generalization of $\varphi\text{-LC}_d$, determined by a (non-empty) set \mathcal{N} of binary modalities and a set σ of structural rules.

Definition A.16 *The set $\mathcal{T}_{\sigma\text{-LC}_d}(V)$ of $\sigma\text{-LC}_d$ types is the smallest set that contains a set V of atomic types and that is closed, for any $n \in \mathcal{N}$, under the binary operations \times_n , $/_n$, \setminus_n , \oplus_n , \otimes_n , and \odot_n , i.e. the elements of $\mathcal{T}_{\sigma\text{-LC}_d}(V)$ are defined inductively in the following way:⁵*

$$\mathcal{T} = V \mid (\mathcal{T} \times_n \mathcal{T}) \mid (\mathcal{T} /_n \mathcal{T}) \mid (\mathcal{T} \setminus_n \mathcal{T}) \mid (\mathcal{T} \oplus_n \mathcal{T}) \mid (\mathcal{T} \otimes_n \mathcal{T}) \mid (\mathcal{T} \odot_n \mathcal{T}).$$

Definition A.17 *A $\sigma\text{-LC}_d$ configuration is an abstract tree with nodes labeled by \circ_n , $>_n$, and $<_n$, and leaves labeled by $\sigma\text{-LC}_d$ types, i.e. $\sigma\text{-LC}_d$ configurations are defined inductively in the following way.⁶*

$$\tau = \mathcal{L} \mid (\tau \circ_n \tau) \mid (\tau >_n \tau) \mid (\tau <_n \tau).$$

Definition A.18 *A $\sigma\text{-LC}_d$ sequent $\sigma \Rightarrow \tau$ is an ordered pair (σ, τ) of $\sigma\text{-LC}_d$ configurations.*

Just as for **LGC**, we can define the notion of duality based, for all modalities n , on the pairs of dual operators $\{\times_n, \oplus_n\}$, $\{/_n, \otimes_n\}$, and $\{\setminus_n, \odot_n\}$. Then the dual of a type is defined as follows:

⁵As usual, outer parenthesis will be dropped.

⁶As usual, outer parenthesis will be dropped.

- if A is atomic, then $A_d = A$; otherwise,
- $(A \diamond B)_d = A_d \diamond_d B_d$ where $\{\diamond, \diamond_d\}$ is any pair of dual operators.

The dual configuration τ_d of a configuration τ is given by the clauses:

- if τ is a type \mathcal{T} , then $\tau_d = \mathcal{T}_d$;
- if $\tau = \sigma \diamond \rho$, then $\tau_d = \sigma_d \diamond_d \rho_d$ for any $\diamond \in \{\circ_n, >_n, <_n\}$.

The dual of an **LGC** sequent $\sigma \Rightarrow \tau$ is $\tau_d \Rightarrow \sigma_d$.

Let $\mathcal{H} = \{H_n : n \in \mathbb{N}\}$ be a set disjoint from $\mathcal{T}_{\sigma\text{-LC}_d}(V)$.

Definition A.19 A $\sigma\text{-LC}_d$ context with m holes ($m \geq 0$) is an abstract tree $\sigma[H_1, \dots, H_m]$ with nodes labeled by $\circ_n, >_n$, and $<_n$, m leaves labeled by H_1, \dots, H_m and the other leaves, if there are any, labeled by $\sigma\text{-LC}_d$ types. A leaf labeled by an element of \mathcal{H} is called a hole, a leaf labeled by an element of \mathcal{L} is said to be proper. A $\sigma\text{-LC}_d$ context is pure if it has no proper leaves.

The notation $\sigma[H_i]$ will be used to denote an m -holed context, when we want to focus on the hole labeled by H_i .

In a $\sigma\text{-LC}_d$ context $\sigma[H_1, \dots, H_m]$, any hole H_i ($i = 1, \dots, m$) has a sign p_i , which can be either positive or negative. If we need to mention explicitly the sign, we write $\sigma[H_1^{p_1}, \dots, H_m^{p_m}]$.

Definition A.20 Let H be a hole in a $\sigma\text{-LC}_d$ context $\sigma[H]$. We say that

- H has positive sign if $\sigma[H]$ is the trivial context H ; or
- H has sign p if $\sigma[H]$ is a non-trivial context that can be written in one of the following ways, where τ is any context, and $p \neq q$ can be either $+$ or $-$:

$$\sigma[H^p] = \begin{array}{l|l|} \sigma[H^p] \circ_n \tau & \tau \circ_n \sigma[H^p] & | \\ \sigma[H^p] <_n \tau & \tau <_n \sigma[H^q] & | \\ \sigma[H^q] >_n \tau & \tau >_n \sigma[H^p] & | \end{array}$$

The result of substituting a context ρ in the H hole of a context $\sigma[H]$ is the context $\sigma[\rho]$ defined as follows, where $\diamond \in \{\circ_n, >_n, <_n\}$ for any modality n :

- if $\sigma[H] = H$, then $\sigma[\rho] = \rho$;
- if $\sigma[H] = \xi[H] \diamond \tau$, then $\sigma[\rho] = \xi[\rho] \diamond \tau$;
- if $\sigma[H] = \tau \diamond \xi[H]$, then $\sigma[\rho] = \tau \diamond \xi[\rho]$.

Definition A.21 A structural rule in $\sigma\text{-LC}_{\mathbf{d}}$ is determined by a context restructuring rule, i.e. two pairs of pure contexts, such as

$$\langle \sigma_1[H_1^{p_1}, \dots, H_m^{p_m}], \tau_1[H_{m+1}^{p_{m+1}}, \dots, H_l^{p_l}] \rangle$$

and

$$\langle \sigma_2[H_1^{p_1}, \dots, H_m^{p_m}], \tau_2[H_{m+1}^{p_{m+1}}, \dots, H_l^{p_l}] \rangle,$$

where, for each hole, its sign in the first and in the second pair coincide. The associated $\sigma\text{-LC}_{\mathbf{d}}$ structural rule is the ordered pair

$$\frac{\sigma_1[\rho_1, \dots, \rho_m] \Rightarrow \tau_1[\rho_{m+1}, \dots, \rho_l]}{\sigma_2[\rho_1, \dots, \rho_m] \Rightarrow \tau_2[\rho_{m+1}, \dots, \rho_l]}$$

where ρ_1, \dots, ρ_l are variables that range over $\sigma\text{-LC}_{\mathbf{d}}$ configurations.

To preserve the symmetry of the calculus, we require that the set σ of structural rules be closed under duality, i.e. if a rule is in σ then its dual belongs to σ as well.

Definition A.22 The rules of the $\sigma\text{-LC}_{\mathbf{d}}$ calculus comprise the minimal set of rules listed in Figure A.3 on page 190 and the set σ of structural rules.

Here is an example of a structural rule

$$\frac{(\tau_1 \circ_n \tau_2) \circ_n \tau_3 \Rightarrow \tau}{\tau_1 \circ_n (\tau_2 \circ_n \tau_3) \Rightarrow \tau}$$

It allows us to prove the theorem $A \times_n (B \times_n C) \Rightarrow (A \times_n B) \times_n C$. To be able to prove its dual, i.e. $(A \oplus_n B) \oplus_n C \Rightarrow A \oplus_n (B \oplus_n C)$ one needs the dual structural rule:

$$\frac{\sigma \Rightarrow (\tau_1 \circ_n \tau_2) \circ_n \tau_3}{\sigma \Rightarrow \tau_1 \circ_n (\tau_2 \circ_n \tau_3)}$$

Observe that, because of the display rule, any structural rule could be written as involving exclusively a restructuring of, e.g., the premises of the sequents. For instance, the associative rule just mentioned above can be written equivalently in the following form:

$$\frac{\tau_1 >_n (\tau_2 <_n \tau_3) \Rightarrow \sigma}{(\tau_1 >_n \tau_2) <_n \tau_3 \Rightarrow \sigma}$$

Not restricting structural rules to affect only one side of a sequent allows us to write rules in a more intuitive way.

Lemma A.23 *For any σ - \mathbf{LC}_d derivation with the following structure:*

$$\frac{\begin{array}{c} \vdots D_1 \\ \tau \Rightarrow A \end{array} \quad \begin{array}{c} \vdots D_2 \\ A \Rightarrow \sigma \end{array}}{\tau \Rightarrow \sigma}$$

where D_1 and D_2 are cut-free, there is a cut-free σ - \mathbf{LC}_d derivation of $\tau \Rightarrow \sigma$.

Proof. The proof is by induction over the complexity of the cut, defined as the number of logical operators that appear in τ , A , and σ . If either $\tau \Rightarrow A$ or $A \Rightarrow \sigma$ is an axiom, there is nothing to show (this covers the basic case of the induction). Otherwise, let r be the last logical rule in \mathcal{D}_1 . If r introduces A , then r is a right rule and, without loss of generality, we can assume that it is the last rule in \mathcal{D}_1 . We will show the case in which r is the \times right rule, the other cases being similar. The derivation we start with has the following form, where $\Sigma[A_1 \times A_2]$ is a possibly empty list of structural rules, \mathcal{E}_1 , \mathcal{E}_2 , and \mathcal{F}_1 are cut-free derivations and $\mathcal{F}_2[A_1 \times A_2]$ is a cut-free partial derivation (in the sense that one of its premises is the sequent $A_1 \times A_2 \Rightarrow \sigma'$):

$$\frac{\begin{array}{c} \vdots \mathcal{E}_1 \\ \tau_1 \Rightarrow A_1 \end{array} \quad \begin{array}{c} \vdots \mathcal{E}_2 \\ \tau_2 \Rightarrow A_2 \end{array} \quad \begin{array}{c} \vdots \mathcal{F}_1 \\ A_1 \circ A_2 \Rightarrow \sigma' \end{array}}{\begin{array}{c} \tau_1 \circ \tau_2 \Rightarrow A_1 \times A_2 \\ \vdots \Sigma[A_1 \times A_2] \\ \tau \Rightarrow A_1 \times A_2 \end{array} \quad \begin{array}{c} \vdots \mathcal{F}_2[A_1 \times A_2] \\ A_1 \times A_2 \Rightarrow \sigma \end{array}}{\tau \Rightarrow \sigma}$$

The following derivation is obtained pasting, by a cut rule, two subderivations of the previous derivation (plus we use once a display rule):

$$\frac{\frac{\frac{\vdash \mathcal{E}_2}{\tau_2 \Rightarrow A_2}}{\tau_2 \Rightarrow A_1 > \sigma'} \quad \frac{\frac{\vdash \mathcal{F}_1}{A_1 \circ A_2 \Rightarrow \sigma'}}{A_2 \Rightarrow A_1 > \sigma'}}{\tau_2 \Rightarrow A_1 > \sigma'}$$

Since the cut that is used has lower complexity, there is by induction hypothesis a cut-free derivation \mathcal{G}_2 of the last sequent $\tau_2 \Rightarrow A_1 > \sigma'$. This derivation can be combined with \mathcal{E}_1 in the following way:

$$\frac{\frac{\frac{\vdash \mathcal{E}_1}{\tau_1 \Rightarrow A_1}}{\tau_1 \Rightarrow \sigma' < \tau_2} \quad \frac{\frac{\vdash \mathcal{G}_2}{\tau_2 \Rightarrow A_1 > \sigma'}}{A_1 \circ \tau_2 \Rightarrow \sigma'}}{\tau_1 \Rightarrow \sigma' < \tau_2}$$

Again, by induction hypothesis there is a cut-free derivation \mathcal{G}_1 of $\tau_1 \Rightarrow \sigma' < \tau_2$. Denote by $\mathcal{F}_2[\tau_1 \circ \tau_2]$ the partial derivation $\mathcal{F}_2[A_1 \times A_2]$, where at each step the type $A_1 \times A_2$ has been substituted by the configuration $\tau_1 \circ \tau_2$. Similarly, let $\Sigma_1[\tau_1 \circ \tau_2]$ be the partial derivation $\Sigma_1[A_1 \times A_2]$ upon substitution of $\tau_1 \circ \tau_2$ for $A_1 \times A_2$. This yields the sought for cut-free derivation of $\tau \Rightarrow \sigma$:

$$\frac{\frac{\frac{\vdash \mathcal{G}_1}{\tau_1 \Rightarrow \sigma' < \tau_2}}{\tau_1 \circ \tau_2 \Rightarrow \sigma'} \quad \frac{\frac{\vdash \mathcal{F}_2[\tau_1 \circ \tau_2]}{\tau_1 \circ \tau_2 \Rightarrow \sigma}}{\tau \Rightarrow \sigma}}{\tau \Rightarrow \sigma}$$

If r introduces a type different from A , the derivation has the following structure, where \mathcal{E}_1 , \mathcal{E}_2 , and \mathcal{D}' are cut-free derivations, and $\Sigma[A]$ is a sequence of display and structural rules (we illustrate the case in which r is a binary rule, the unary case being similar):

$$\begin{array}{c}
\frac{\frac{\frac{\vdots \mathcal{E}_1}{\tau_3 \Rightarrow \tau_4}}{\tau_1 \Rightarrow \tau_2}}{\frac{\frac{\frac{\vdots \mathcal{E}_2}{\tau_5 \Rightarrow \tau_6}}{\tau \Rightarrow A}}{A \Rightarrow \sigma}} \\
\tau \Rightarrow \sigma
\end{array}$$

Now, A must appear in the sequent conclusion and in one of the sequent premises of r . To fix ideas, say that it appears in $\tau_2 = \tau_2[A]$ and in $\tau_5 = \tau_5[A]$ (the other cases are similar). Then, for a suitable sequence $\Delta[A]$ of display rules, we can consider the following derivation:

$$\begin{array}{c}
\frac{\frac{\frac{\vdots \mathcal{E}_2}{\tau_5[A] \Rightarrow \tau_6}}{\tau'_5 \Rightarrow A}}{A \Rightarrow \sigma_2} \\
\tau'_5 \Rightarrow \sigma
\end{array}$$

Since the last rule in the derivation above is a cut of complexity lower than the cut in the initial derivation, by induction hypothesis there is a cut-free derivation \mathcal{G} of $\tau'_5 \Rightarrow \sigma$. Let $\Delta^\sharp[A]$ be the sequence $\Delta[A]$ of display rules in inverse order and let $\Delta^\sharp[\sigma]$ be $\Delta^\sharp[A]$ where, at every step, the type A has been substituted by the configuration σ . This allows us to combine \mathcal{G} and \mathcal{E}_1 into the following cut-free derivation of $\tau \Rightarrow \sigma$:

$$\begin{array}{c}
\frac{\frac{\frac{\frac{\vdots \mathcal{E}_1}{\tau_3 \Rightarrow \tau_4}}{\tau_1 \Rightarrow \tau_2[\sigma]}}{\Sigma[\sigma]}}{\tau \Rightarrow \sigma} \\
\frac{\frac{\frac{\vdots \mathcal{G}}{\tau'_5 \Rightarrow \sigma}}{\tau_5[\sigma] \Rightarrow \tau_6}}{\tau_5[\sigma] \Rightarrow \tau_6}
\end{array}$$

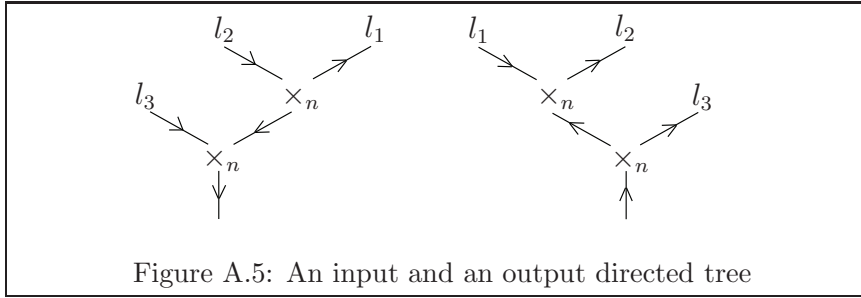
All cases having been considered, the result is established. \square

Given the previous lemma, a straightforward proof by induction over the number of cuts in a derivation establishes the following result.

Theorem A.24 (Cut Elimination Property) *Any $\sigma\text{-LC}_d$ theorem has a cut-free derivation.*

A.4 Proof net theory for $\sigma\text{-LC}_d$

Definition A.25 *A $\sigma\text{-LC}_d$ tree is an acyclic and connected directed graph with unary and ternary nodes. Ternary nodes, if there are any, are labeled by elements of the set $\{\times_n : n \in \mathcal{N}\}$ and come with a cyclic order on the incident edges. At any ternary node, at least one adjacent edge points toward the node and one, away from it. One unary node is called the root and is not labeled; the other unary nodes, of which there is at least one, are called the leaves and are labeled by elements of \mathcal{L} .*



If the edge adjacent to the root points toward it, we speak of an input tree; if it points away from it, we speak of an output tree (see Figure A.5).

Definition A.26 *The set of polar $\sigma\text{-LC}_d$ trees is the smallest set defined by double induction in the following way:⁷*

$$\begin{aligned} \text{input trees: } \mathcal{I} &= \mathcal{L}^\bullet \mid (\mathcal{I} \circ_n \mathcal{I}) \mid (\mathcal{I} <_n \mathcal{O}) \mid (\mathcal{O} >_n \mathcal{I}) \\ \text{output trees: } \mathcal{O} &= \mathcal{L}^\circ \mid (\mathcal{O} \circ_n \mathcal{O}) \mid (\mathcal{I} >_n \mathcal{O}) \mid (\mathcal{O} <_n \mathcal{I}) \end{aligned}$$

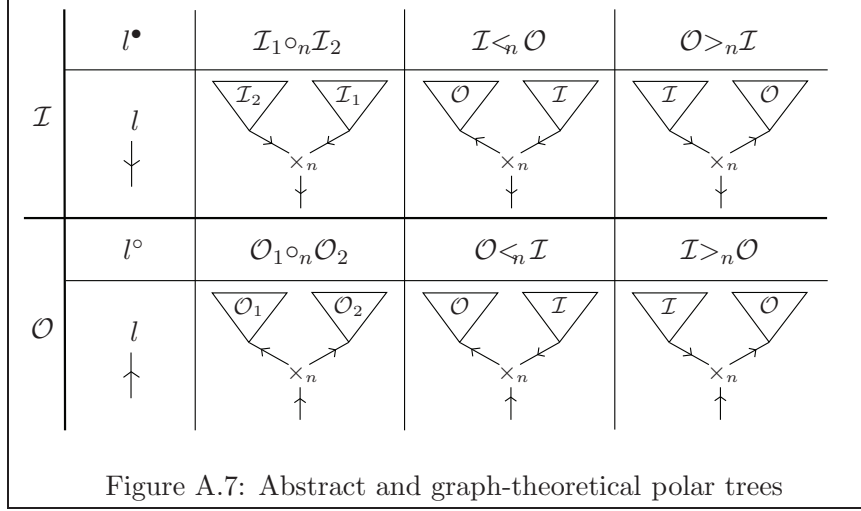
The inductive definition of Figure A.6 assigns to any configuration an input and an output $\sigma\text{-LC}_d$ tree.

The correspondence between polar trees and their graph-theoretical representation as $\sigma\text{-LC}_d$ trees is illustrated in Figure A.7.

⁷As usual, outer parenthesis will be dropped.

configuration	l	$\tau \circ_n \sigma$	$\tau \triangleleft_n \sigma$	$\sigma \triangleright_n \tau$
input tree	l^\bullet	$\tau^\bullet \circ_n \sigma^\bullet$	$\tau^\bullet \triangleleft_n \sigma^\circ$	$\sigma^\circ \triangleright_n \tau^\bullet$
output tree	l°	$\sigma^\circ \circ_n \tau^\circ$	$\sigma^\bullet \triangleright_n \tau^\circ$	$\tau^\circ \triangleleft_n \sigma^\bullet$

Figure A.6: Configurations and polar trees



A σ - \mathbf{LC}_d tree context is obtained, as in the previous sections, enriching the set of leaves with a disjoint set $\{H_i : i \in \mathbb{N}\}$ of hole labels. A hole H_i is an input (output) hole if the adjacent edge points away (toward) the node labeled by H_i . Substitution of a tree in a context carries over with one minor restriction on polarity: an input (output) tree can be substituted only for an input (output) hole.

Consider the set of pairs $(\mathcal{I}, \mathcal{O})$ of an input and an output σ - \mathbf{LC}_d context. On this set let \approx be the smallest equivalence relation such that for all $n \in \mathcal{N}$, for all input contexts $\mathcal{I}_0, \mathcal{I}_1$, and output contexts \mathcal{O}_1 and \mathcal{O}_2 :

- $(\mathcal{I}_1, \mathcal{O}_1 \triangleleft_n \mathcal{I}_2) \approx (\mathcal{I}_1 \circ_n \mathcal{I}_2, \mathcal{O}_1) \approx (\mathcal{I}_2, \mathcal{I}_1 \triangleright_n \mathcal{O}_1)$; and
- $(\mathcal{I}_1 \triangleleft_n \mathcal{O}_2, \mathcal{O}_1) \approx (\mathcal{I}_1, \mathcal{O}_1 \circ_n \mathcal{O}_2) \approx (\mathcal{O}_1 \triangleright_n \mathcal{I}_1, \mathcal{O}_2)$.

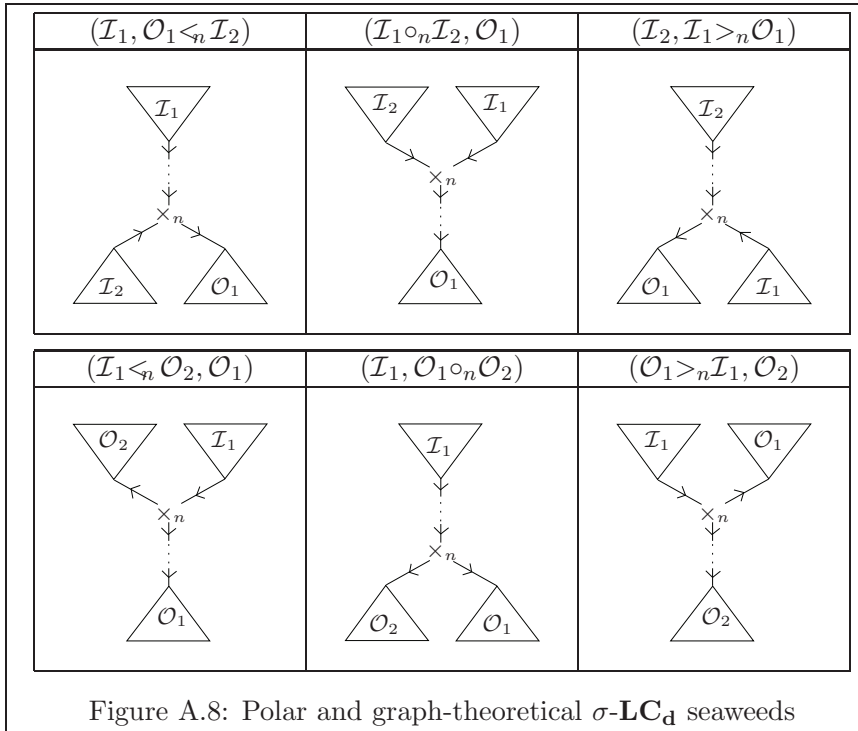
Denote by $\mathcal{I} \star \mathcal{O}$ the \approx -equivalence class individuated by the pair $(\mathcal{I}, \mathcal{O})$.

Definition A.27 A polar σ - \mathbf{LC}_d seaweed is an \approx -equivalence class.

Definition A.28 A $\sigma\text{-LC}_d$ seaweed context is an acyclic and connected directed graph with unary and ternary nodes. The ternary nodes are labeled by elements of the set $\{\times_n : n \in \mathcal{N}\}$; the unary nodes called (proper) leaves are labeled by elements of \mathcal{L} ; the unary nodes called holes are labeled by (distinct) elements of $\mathcal{H} = \{H_i : i \in \mathbb{N}\}$. It has no other nodes and at least one edge. Ternary nodes come with a cyclic order on the incident edges; at any ternary node, at least one of the incident edges points toward the node and at least one points away from it.

Definition A.29 A $\sigma\text{-LC}_d$ seaweed context with no holes is called a $\sigma\text{-LC}_d$ seaweed. A $\sigma\text{-LC}_d$ seaweed context with no proper leaves is said to be pure.

The correspondence between a polar $\sigma\text{-LC}_d$ seaweed and its graph-theoretical representation is illustrated, for non-trivial cases, in Figure A.8.



Consider the following sets:

- $\mathcal{N}_\times = \{\times_i : i \in \mathcal{N}\}$ of *tensor operators*;
- $\mathcal{N}_\wp = \{\wp_i : i \in \mathcal{N}\}$ of *par operators*.

Definition A.30 A $\sigma\text{-LC}_d$ partial proof structure is a connected directed graph such that:

- its edges are labeled by $\sigma\text{-LC}_d$ types;
- its ternary nodes are labeled by elements of $\mathcal{N}_\times \cup \mathcal{N}_\emptyset$; incident edges are ordered cyclically; one of them is the conclusion, the left (right) premise is the edge that immediately follows (precedes) the conclusion in clockwise fashion; the premises are labeled by the immediate subformulas of the type labeling the conclusion; at any ternary node, at least one of the edges points toward the node and at least one, away from it;
- its binary nodes are labeled by ID and CUT (or more simply, represented by a horizontal line); ID nodes have two conclusions, CUT nodes have two premises; at any binary node, the incident edges have identical labels; one of the edges points toward the node, while the other edge points away from it;
- unary nodes are not labeled;
- there are no other nodes;
- any edge is the premise of at most one node and the conclusion of at most one node.

For any $l \in \mathcal{N}_\times \cup \mathcal{N}_\emptyset \cup \{ID, CUT\}$, a node labeled by l together with its incident edges is called an l -link. We speak of an input (output) logical link when the conclusion points away from (toward) the node of the link. Labels and directions in logical links are related in the way exhibited in Figure A.9. If $l \in \mathcal{N}_\emptyset$ the link is said to be of type i or, equivalently, to be a par link. If $l \in \mathcal{N}_\times$, the link is said to be of type ii or, equivalently, a tensor link.

Definition A.31 An open leaf in a $\sigma\text{-LC}_d$ partial proof structure is an edge that is the conclusion of no link.

Definition A.32 A $\sigma\text{-LC}_d$ proof structure is a $\sigma\text{-LC}_d$ partial proof structure with no open leaves.

Definition A.33 A $\sigma\text{-LC}_d$ pseudostructure is a connected directed graph such that:

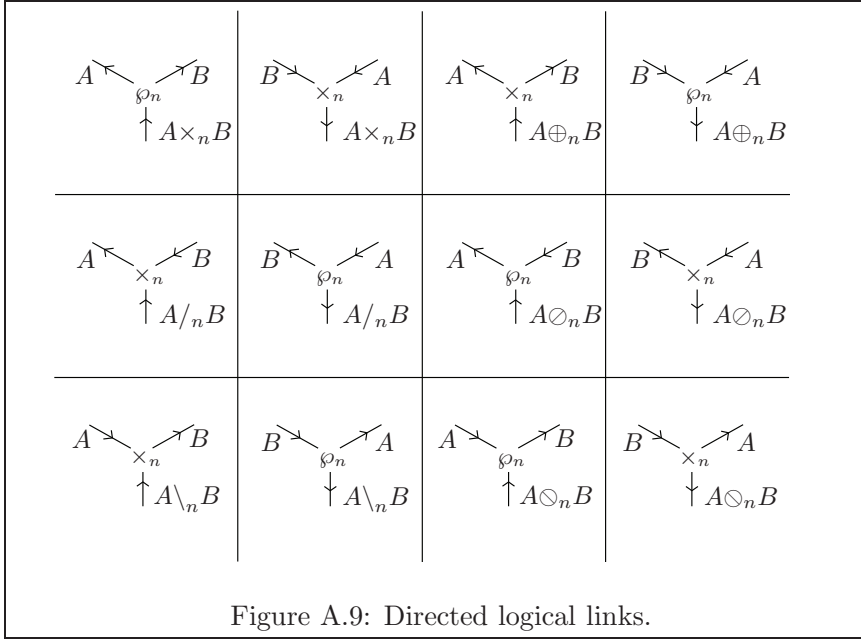


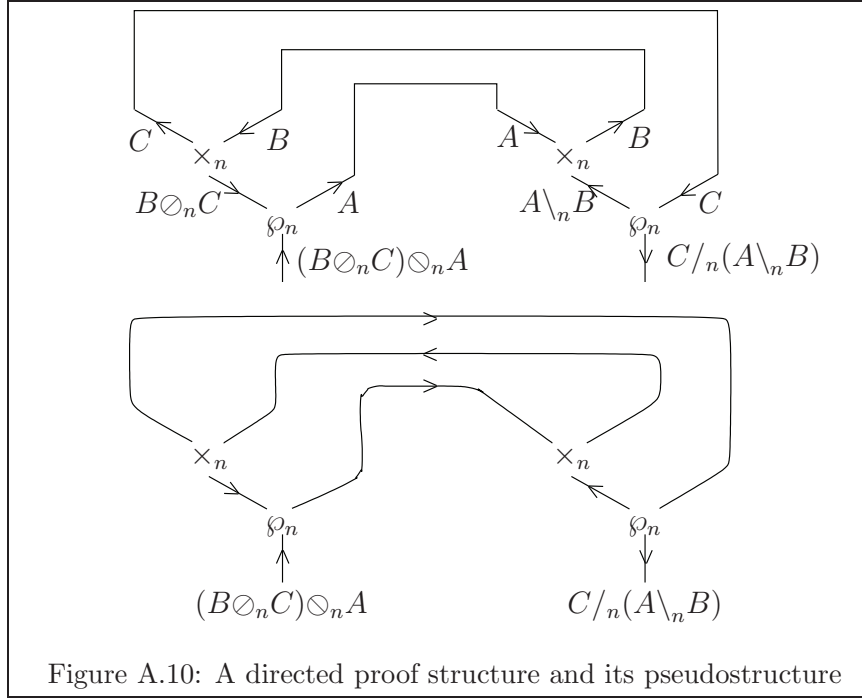
Figure A.9: Directed logical links.

- its ternary nodes are labeled by elements of $\mathcal{N}_\times \cup \mathcal{N}_\varphi$; incident edges are ordered cyclically; in the case of the nodes labeled by elements of \mathcal{N}_φ , one of the edges is specified to be the conclusion (and denoted by a tail on the edge); at any ternary node, at least one of the edges points toward the node and one, away from it;
- unary nodes are labeled by types of $\sigma\text{-LCA}_d$;
- there are no other nodes;
- edges are not labeled.

In a pseudostructure, we can still speak of φ -links as the nodes labeled by an $n \in \mathcal{N}_\varphi$ together with its incident edges.

Given any directed partial proof structure Π , the *underlying* directed pseudostructure Π^- is obtained in the following way (see Figure A.10):

- for any edge incident on a unary node, copy the label of the edge onto the node itself;
- mark the conclusion of any node labeled by an element of \mathcal{N}_φ with a tail;



- remove edge labels;
- remove any binary node by connecting directly between them the two incident edges; the new edge inherits the direction of the two joined edges.

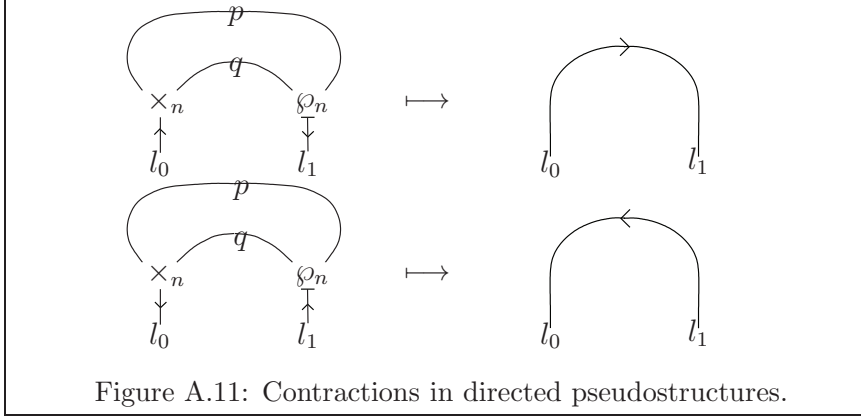
The contraction rule (CON_n) for $\sigma\text{-LC}_d$ pseudostructures can have any of the forms pictured in Figure A.11, where p and q stand for any direction on the edges.

Any structural rule

$$\frac{\sigma_1[\rho_1, \dots, \rho_m] \Rightarrow \tau_1[\rho_{m+1}, \dots, \rho_l]}{\sigma_2[\rho_1, \dots, \rho_m] \Rightarrow \tau_2[\rho_{m+1}, \dots, \rho_l]}$$

induces the rewriting rule on $\sigma\text{-LC}_d$ pseudostructures determined by (the graphical representation of) the following modification:

$$\sigma_1[l_1, \dots, l_m]^\bullet \star \tau_1[l_{m+1}, \dots, l_l]^\circ \rightarrow \sigma_2[l_1, \dots, l_m]^\bullet \star \tau_2[l_{m+1}, \dots, l_l]^\circ.$$



Let \mathcal{R}' be the set of rewriting rules for pseudostructures derived from the set σ of structural rules of $\sigma\text{-LC}_{\mathbf{d}}$. Let \mathcal{R} be the set \mathcal{R}' enriched with the set of contraction rules (CON_n) , for any modality $n \in \mathcal{N}$. Given any two $\sigma\text{-LC}_{\mathbf{d}}$ pseudostructures Π and Σ , say that $\Sigma <_{\mathcal{R}} \Pi$ if Π can be rewritten as Σ applying a rule of \mathcal{R} . Let $\leq_{\mathcal{R}}$ be the reflexive-transitive closure of $<_{\mathcal{R}}$.

Definition A.34 A $\sigma\text{-LC}_{\mathbf{d}}$ pseudostructure Π is \mathcal{R} -contractible if there is a $\sigma\text{-LC}_{\mathbf{d}}$ seaweed α such that $\alpha \leq_{\mathcal{R}} \Pi$. A $\sigma\text{-LC}_{\mathbf{d}}$ proof structure is \mathcal{R} -contractible if the underlying pseudostructure Π^- is \mathcal{R} -contractible.

Proposition A.35 If \mathcal{D} is a $\sigma\text{-LC}_{\mathbf{d}}$ derivation of a sequent $\sigma \Rightarrow \tau$, then $(\mathcal{D})^-$ rewrites as $\sigma^\bullet \star \tau^\circ$.

Proof. By induction over the length n of \mathcal{D} . If $n = 0$, i.e. if α is an axiom, there is nothing to prove. If $n \geq 1$ consider the last rule r in \mathcal{D} . If r is an instance of the left \times_n rule

$$\frac{\frac{\vdots D_0}{A \circ_n B \Rightarrow \sigma}}{A \times_n B \Rightarrow \sigma}$$

then by induction hypothesis $(\mathcal{D}_0)^-$ rewrites as $(A \circ_n B)^\bullet \star \sigma^\circ$. The same rewriting rules applied to $(\mathcal{D})^-$ yield the graph obtained joining in $(A \circ_n B)^\bullet \star \sigma^\circ$ the leaves labeled by A and B by the input link with conclusion $A \times_n B$. A further contraction yields a seaweed equivalent to $(A \times_n B)^\bullet \star \sigma^\circ$. If r is an instance of any other unary logical rule, the reasoning is similar.

If the last rule r used in \mathcal{D} is an instance of the right \times_n rule

$$\frac{\frac{\vdash D_0}{\tau_0 \Rightarrow A_0} \quad \frac{\vdash D_1}{\tau_1 \Rightarrow A_1}}{\tau_0 \circ_n \tau_1 \Rightarrow A_0 \times_n A_1}$$

then by induction hypothesis $(\mathcal{D}_i)^-$, following a list ρ_i of rewriting rules, rewrites as $\tau_i^\bullet \star A_i^\circ$ for $i \in \{0, 1\}$. Following the lists ρ_1 and ρ_2 , $(\mathcal{D})^-$ yields $(\tau_0 \circ_n \tau_1)^\bullet \star (A_0 \times_n A_1)^\circ$. The other logical binary cases are similar.

If \mathcal{D} consists of a derivation \mathcal{D}' followed by an instance of a structural rule s , apply the induction hypothesis to $(\mathcal{D}')^-$ and rewrite the resulting sequent applying the rewriting rule that corresponds to s . As for the display rules, recall that $\sigma\text{-LC}_{\mathbf{d}}$ seaweeds are invariant for them. The proof is completed observing that the case of cut is analogous to the tensor case (with the minor difference that no further contraction is needed). \square

The same reasoning as in Lemma 1.24 proves the following result.

Lemma A.36 (The Bridge Splitting Lemma) *Let Π be an $\sigma\text{-LC}_{\mathbf{d}}$ contractible \mathcal{NL} structure such that:*

- *it has at least a non-final \wp -link; and*
- *no \wp -link is final.*

Then there is a bridging \wp - with conclusion X that splits Π into a contractible $\sigma\text{-LC}_{\mathbf{d}}$ structure Π_0 and a contractible $\sigma\text{-LC}_{\mathbf{d}}$ partial structure $\Pi_1(X)$.

Theorem A.37 (Sequentialization) *Let Π be a $\sigma\text{-LC}_{\mathbf{d}}$ proof structure such that Π^- rewrites as a $\sigma^\bullet \star \tau^\circ$. Then there is a $\sigma\text{-LC}_{\mathbf{d}}$ derivation \mathcal{D} of $\sigma \Rightarrow \tau$ such that $(\mathcal{D}) = \Pi$.*

Proof. By induction over the complexity n of Π . If $n = 0$ there is nothing to show. Suppose Π has a final \wp -node L , the conclusion of which is labeled by X . The removal of L yields a proof structure Π' such that Π'^- is contractible. By induction hypothesis, Π' can be sequentialized as a derivation \mathcal{D}' . Completing \mathcal{D}' with a unary logical rule that introduces X as main formula yields a derivation \mathcal{D} of $\sigma \Rightarrow \tau$ such that $\Pi = (\mathcal{D})$. If Π has no final \wp -node, then, by the previous Splitting Lemma, there is a bridging \wp -link that decomposes Π into a proof structure Π_0 and a partial proof structure $\Pi_1(X)$ such that Π_0^- and $\Pi_1(X)^-$ are contractible. Note that $\Pi_1(X)$ can be turned into a proof structure Π_1 simply by adding an identity link with

conclusion X . $\Pi_1(X)$ and Π_1 coincide qua unlabeled graphs and therefore also Π_1^- is contractible. By induction hypothesis, there are derivations \mathcal{D}_0 and \mathcal{D}_1 such that $(\mathcal{D}_0) = \Pi_0$ and $(\mathcal{D}_1) = \Pi_1$. Replacing in the derivation \mathcal{D}_1 the axiom $X \Rightarrow X$ with the derivation \mathcal{D}_0 yields a derivation \mathcal{D} of $\sigma^\bullet \star \tau^\circ$ such that $(\mathcal{D}) = \Pi$. \square

A.5 Displaying LGC

Displaying **LGC** yields a calculus **LGC_d** that shares the same set of types and configurations. Its rules comprise the minimal set of rules of Figure A.3 and the set of structural rules of Figure A.12.

(G1) $\frac{\rho > (\sigma \circ \tau) \Rightarrow \psi}{(\rho > \sigma) \circ \tau \Rightarrow \psi}$	(G3) $\frac{(\rho \circ \sigma) < \tau \Rightarrow \psi}{\rho \circ (\sigma < \tau) \Rightarrow \psi}$
(G2) $\frac{\rho > (\sigma \circ \tau) \Rightarrow \psi}{\sigma \circ (\rho > \tau) \Rightarrow \psi}$	(G4) $\frac{(\rho \circ \sigma) < \tau \Rightarrow \psi}{(\rho < \tau) \circ \sigma \Rightarrow \psi}$

Figure A.12: Structural rules of **LGC_d**.

Corollary A.38 *LGC_d enjoys the Cut Elimination Property.*

Observe that the set of structural rules of **LGC_d** can be equivalently expressed by their dual rules listed in Figure A.13. Note, however, that the dual of (G1) is equivalent to (G3) (and viceversa), whereas the dual of (G2) is equivalent to (G2) itself (and similarly for the (G4) rule).

Definition A.39 *A \circ -context is an LGC_d context in which the structural operators $>$ and $<$ do not appear.*

Definition A.40 *The anticontext $\tilde{\sigma}[K]$ of the \circ -context $\sigma[H]$ is defined by induction in the following way:*

(G1') $\frac{\psi \Rightarrow \rho > (\sigma \circ \tau)}{\psi \Rightarrow (\rho > \sigma) \circ \tau}$	(G3') $\frac{\psi \Rightarrow (\rho \circ \sigma) < \tau}{\psi \Rightarrow \rho \circ (\sigma < \tau)}$
(G2') $\frac{\psi \Rightarrow \rho > (\sigma \circ \tau)}{\psi \Rightarrow \sigma \circ (\rho > \tau)}$	(G4') $\frac{\psi \Rightarrow (\rho \circ \sigma) < \tau}{\psi \Rightarrow (\rho < \tau) \circ \sigma}$

Figure A.13: Alternative set of structural rules of **LGC_d**.

- if $\sigma[H] = H$, then $\tilde{\sigma}[K] = K$;
- if $\sigma[H] = \tau \circ \sigma[H]$, then $\tilde{\sigma}[K] = \tilde{\sigma}[\tau > K]$;
- if $\sigma[H] = \sigma[H] \circ \tau$, then $\tilde{\sigma}[K] = \tilde{\sigma}[K < \tau]$.

A simple induction over the structural complexity of a \circ -context establishes the following result.

Lemma A.41 *For any \circ -context $\sigma[H]$ and any configurations τ and ρ , the sequent $\sigma[\tau] \Rightarrow \rho$ is an \mathbf{LGC}_d theorem if and only if $\tau \Rightarrow \tilde{\sigma}[\rho]$ is an \mathbf{LGC}_d theorem. Dually, the sequent $\rho \Rightarrow \sigma[\tau]$ is an \mathbf{LGC}_d theorem if and only if $\tilde{\sigma}[\rho] \Rightarrow \tau$ is an \mathbf{LGC}_d theorem.*

Lemma A.42 *For any \circ -context $\tau[H]$, the following inferences are valid in \mathbf{LGC}_d :*

$\frac{\sigma \Rightarrow B > \tau[A]}{\sigma \Rightarrow \tau[B > A]}$	$\frac{\sigma \Rightarrow \tau[A] < B}{\sigma \Rightarrow \tau[A < B]}$
$\frac{B > \tau[A] \Rightarrow \sigma}{\tau[B > A] \Rightarrow \sigma}$	$\frac{\tau[A] < B \Rightarrow \sigma}{\tau[A < B] \Rightarrow \sigma}$

Proof. The proof is by induction over the structural complexity of $\tau[\]$. In the base case there is nothing to show. Suppose we start with the sequent $\sigma \Rightarrow B > (\tau_1[A] \circ \tau_2)$. Then, using $(G1')$, we obtain $\sigma \Rightarrow (B > \tau_1[A]) \circ \tau_2$ and from this, by induction hypothesis, we can derive $\sigma \Rightarrow \tau_1[B > A] \circ \tau_2$. All the other cases are similar. \square

Lemma A.43 *Any \mathbf{LGC} rule is a valid inference rule in \mathbf{LGC}_d .*

Proof. We only need to check the cases in which the active formula of the \mathbf{LGC} rule is embedded in a (non-trivial) context. There are two classes of such cases. In the first one, exemplified here by the \mathbf{LGC} left introduction rule for the tensor, we use Lemma A.41 in the first and last step:

$$\frac{\frac{\frac{\sigma[A \circ B] \Rightarrow \tau}{A \circ B \Rightarrow \tilde{\sigma}[\tau]}}{A \times B \Rightarrow \tilde{\sigma}[\tau]}}{\sigma[A \times B] \Rightarrow \tau}$$

As for the other class, exemplified here by the **LGC** right introduction rule of the operator \setminus , we use Lemma A.42 in the second step and then the Lemma A.41 twice to be able to apply the **LGC_d** right rule for the operator \setminus :

$$\frac{\frac{\frac{B \circ \sigma \Rightarrow \tau[A]}{\sigma \Rightarrow B > \tau[A]}}{\sigma \Rightarrow \tau[B > A]}}{\tilde{\tau}[\sigma] \Rightarrow B > A}}{\tilde{\tau}[\sigma] \Rightarrow B \setminus A}}{\sigma \Rightarrow \tau[B \setminus A]}$$

□

A proof by induction over the length of **LGC** derivations establishes, in the light of the previous lemma, the following result.

Proposition A.44 *All **LGC** theorems are **LGC_d** theorems.*

Definition A.45 *For any **LGC_d** configuration σ , the left closure $l\sigma$ and the right closure $r\sigma$ are defined by double induction in the following way:*

- if $\sigma = A$ is a trivial configuration, then $lA = rA = A$;
- if $\sigma = \tau \circ \rho$, then $l\sigma = l\tau \times l\rho$ and $r\sigma = r\tau \oplus r\rho$;
- if $\sigma = \tau > \rho$, then $l\sigma = r\tau \odot l\rho$ and $r\sigma = l\tau \setminus r\rho$;
- if $\sigma = \tau < \rho$, then $l\sigma = l\tau \odot r\rho$ and $r\sigma = r\tau / l\rho$.

Lemma A.46 *For any **LGC_d** configuration σ , the sequents $\sigma \Rightarrow l\sigma$ and $r\sigma \Rightarrow \sigma$ are **LGC_d** theorems.*

Proof. As for Lemma A.8, the proof is by induction over the structural complexity of the configuration σ , except that now there are more cases to be considered in the inductive step. For instance, if $\sigma = \sigma_1 < \sigma_2$, then by induction hypothesis there are derivations of $\sigma_1 \Rightarrow l\sigma_1$ and $r\sigma_2 \Rightarrow \sigma_2$. Applying the right introduction rule for \odot yields $\sigma_1 < \sigma_2 \Rightarrow l(\sigma_1 < \sigma_2)$, since $l(\sigma_1 < \sigma_2) = l\sigma_1 \odot r\sigma_2$. The other cases are similar. □

Lemma A.47 *For any **LGC_d** type A , the sequent $A \Rightarrow A$ is an **LGC_d** theorem.*

Proof. The usual proof by induction over the complexity of the type A works. For instance, if $A = B \circledast C$, then by induction hypothesis there are derivations of $B \Rightarrow B$ and $C \Rightarrow C$. Applying the right and, then, the left introduction rule for \circledast yields $B < C \Rightarrow B \circledast C$ and hence $B \circledast C \Rightarrow B \circledast C$. \square

Corollary A.48 *For any \mathbf{LGC}_d configuration σ , the sequents $l\sigma \Rightarrow l\sigma$ and $r\sigma \Rightarrow r\sigma$ are \mathbf{LGC}_d theorems.*

Lemma A.49 *If $\frac{\sigma_1 \Rightarrow \tau_1}{\sigma_2 \Rightarrow \tau_2}$ is an \mathbf{LGC}_d rule, then $\frac{l\sigma_1 \Rightarrow r\tau_1}{l\sigma_2 \Rightarrow r\tau_2}$ is a valid \mathbf{LGC} inference. Similarly, if $\frac{\sigma_1 \Rightarrow \tau_1 \quad \sigma_2 \Rightarrow \tau_2}{\sigma_3 \Rightarrow \tau_3}$ is an \mathbf{LGC}_d rule, then $\frac{l\sigma_1 \Rightarrow r\tau_1 \quad l\sigma_2 \Rightarrow r\tau_2}{l\sigma_3 \Rightarrow r\tau_3}$ is a valid \mathbf{LGC} inference.*

Proof. For logical unary rules the result is trivial. For logical binary rules, we will check the case of the right \mathbf{LGC}_d rule for the operator \circledast , the other cases being similar. Recall the rule:

$$\frac{B \Rightarrow \tau \quad \sigma \Rightarrow A}{\sigma < \tau \Rightarrow A \circledast B}$$

In the translation under l and r of the antecedent sequents of the rule, observe that $lB = B$, and $rA = A$. Then:

$$\begin{array}{c} \circledast_R \\ \circledast_L \end{array} \frac{\frac{lB \Rightarrow r\tau \quad l\sigma \Rightarrow rA}{l\sigma \Rightarrow (A \circledast B) \circ r\tau}}{l\sigma \circledast r\tau \Rightarrow A \circledast B}$$

The last line is the translation of the conclusion of the rule, since $r(A \circledast B) = A \circledast B$ and $l(\sigma < \tau) = l\sigma \circledast r\tau$.

For the case of the display rules, we use Corollary A.48 combined with the cut rule. For instance, consider the case of the display rule:

$$\frac{\tau \circ \sigma \Rightarrow \rho}{\sigma \Rightarrow \tau > \rho}$$

In this case the cut will be over $l(\tau \circ \sigma) = l\tau \times l\sigma$. Then:

$$\begin{array}{c} \times_R \\ \text{cut} \\ \backslash_R \end{array} \frac{\frac{\text{Cor. A.48}}{l\tau \Rightarrow l\tau} \quad \frac{\text{Cor. A.48}}{l\sigma \Rightarrow l\sigma}}{l\tau \circ l\sigma \Rightarrow l\tau \times l\sigma} \quad \frac{l(\tau \circ \sigma) \Rightarrow r\rho}{l\tau \circ l\sigma \Rightarrow r\rho}}{l\sigma \Rightarrow l\tau \backslash r\rho}$$

The last line is the translation under l and r of the conclusion of the rule, since $l\tau \setminus r\rho = r(\tau > \rho)$.

For the Grishin structural rule, we need to use the Grishin interaction encoded in the suitable rule for the logical operators. Consider, for instance, rule (G1):

$$\frac{\rho > (\sigma \circ \tau) \Rightarrow \psi}{(\rho > \sigma) \circ \tau \Rightarrow \psi}$$

The Grishin interaction that is needed is encoded in the \odot_L rule. This yields the following **LGC** derivation, where the cut is over the type $r\rho \odot (l\sigma \times l\tau) = l(\rho > (\sigma \circ \tau))$:

$$\begin{array}{c} \begin{array}{c} \text{Cor. A.48} \\ \hline l\sigma \Rightarrow l\sigma \\ \hline \end{array} \quad \begin{array}{c} \text{Cor. A.48} \\ \hline l\tau \Rightarrow l\tau \\ \hline \end{array} \quad \begin{array}{c} \text{Cor. A.48} \\ \hline r\rho \Rightarrow r\rho \\ \hline \end{array} \\ \times_R \frac{\quad}{l\sigma \circ l\tau \Rightarrow l\sigma \times l\tau} \quad \frac{\quad}{r\rho \Rightarrow r\rho} \\ \odot_R \frac{\quad}{l\sigma \circ l\tau \Rightarrow r\rho \circ (r\rho \odot (l\sigma \times l\tau))} \\ \odot_L \frac{\quad}{(r\rho \odot l\sigma) \circ l\tau \Rightarrow r\rho \odot (l\sigma \times l\tau)} \\ \times_L \frac{\quad}{(r\rho \odot l\sigma) \times l\tau \Rightarrow r\rho \odot (l\sigma \times l\tau)} \quad l(\rho > (\sigma \circ \tau)) \Rightarrow r\psi \\ \text{cut} \frac{\quad}{(r\rho \odot l\sigma) \times l\tau \Rightarrow r\psi} \end{array}$$

The last line is the translation under l and r of the conclusion of the rule, since $(r\rho \odot l\sigma) \times l\tau = l((\rho > \sigma) \circ \tau)$.

The proof is completed observing that the case of the cut rule is trivial, since for any cut formula A one has that $lA = rA = A$. \square

A straightforward proof by induction over the length of **LGC_d** derivations establishes, because of the previous lemma, the following result.

Corollary A.50 *If $\sigma \Rightarrow \tau$ is an **LGC_d** theorem, then $l\sigma \Rightarrow r\tau$ is an **LGC** theorem.*

Theorem A.51 *Any **LGC** sequent is a theorem of **LGC** if and only if it is a theorem of **LGC_d**.*

Proof. The ‘only if’ part is Proposition A.44. As for the ‘if’ implication, let the **LGC** sequent $\sigma \Rightarrow \tau$ be an **LGC_d** theorem. By Corollary A.50, $l\sigma \Rightarrow r\tau$ is an **LGC** theorem and therefore, by Proposition A.10, $\sigma \Rightarrow \tau$ is an **LGC** theorem. \square

Theorem A.52 ***LGC_d** is a conservative extension of **NLC**.*

Proof. Clearly, any **NLC** theorem is an **LGC_d** theorem, since all the Lambek rules are valid inferences in **LGC_d**. Now, suppose $\Gamma \Rightarrow C$ is an **NLC** sequent that can be proved in **LGC_d**. We will show that it can be proved in **NLC**. The proof is by induction over the number n of logical rules in cut-free **LGC_d** derivations. If $n = 0$, there is nothing to prove. Suppose $n \geq 1$ and look for the last logical rule r used in the derivation. If r is an instance of a logical rule on the right, say for example the rule for the backslash, then r is the last rule of the derivation, since neither Grishin nor display rules can take place below r . Then we are in the situation reported below on the left:

$$\frac{\begin{array}{c} \vdots D \\ \tau \Rightarrow B > A \end{array}}{\tau \Rightarrow B \setminus A} \qquad \frac{\begin{array}{c} \vdots D \\ \tau \Rightarrow B > A \end{array}}{B \circ \tau \Rightarrow A}$$

The derivation on the right above has fewer logical rules than the former derivation. Hence, by induction hypothesis, the final sequent can be derived in **NLC** and, therefore, $\tau \Rightarrow B \setminus A$ can be derived as well.

If the last logical rule r in the derivation is an instance of a left logical rule, then there are possibly some structural rules below r . They cannot be Grishin rules, since the structure of the final sequent involves only the \circ operator. Therefore they can only be the display rules needed to display the main type of r . If r is an instance of the \times rule, the derivation has the form reported below on the left:

$$\frac{\begin{array}{c} \vdots D \\ A \circ B \Rightarrow \tilde{\sigma}[C] \\ A \times B \Rightarrow \tilde{\sigma}[C] \end{array}}{\sigma[A \times B] \Rightarrow C} \qquad \frac{\begin{array}{c} \vdots D \\ A \circ B \Rightarrow \tilde{\sigma}[C] \end{array}}{\sigma[A \circ B] \Rightarrow C}$$

The derivation on the right has a lower complexity and therefore, by induction hypothesis, $\sigma[A \circ B] \Rightarrow C$ can be derived in **NLC** and so can the sequent $\sigma[A \times B] \Rightarrow C$. If r is an instance of an implication, say the backslash, the derivation has the form reported below on the left:

$$\frac{\begin{array}{c} \vdots D_1 \quad \vdots D_2 \\ A \Rightarrow \psi \quad \phi \Rightarrow B \\ \hline B \setminus A \Rightarrow \phi > \psi \\ \hline \vdots \\ \hline \sigma[B \setminus A] \Rightarrow C \end{array}}{\sigma[B \setminus A] \Rightarrow C} \qquad \frac{\begin{array}{c} \vdots D_1 \quad \vdots D_2 \\ A \Rightarrow \tilde{\xi}[C] \quad \phi \Rightarrow B \\ \hline B \setminus A \Rightarrow \phi > \tilde{\xi}[C] \\ \hline \phi \circ B \setminus A \Rightarrow \tilde{\xi}[C] \\ \hline \xi[\phi \circ B \setminus A] \Rightarrow C \end{array}}{\xi[\phi \circ B \setminus A] \Rightarrow C}$$

Since the rules below r are only instances of the display rules, there are a \circ -configuration ϕ and a \circ -context $\xi[\]$ such that $\sigma[B \setminus A] = \xi[\phi \circ B \setminus A]$ and

$\psi = \tilde{\xi}[C]$, i.e. we can rewrite the \mathbf{LGC}_d derivation above on the left as the derivation above on the right. By induction hypothesis, there are \mathbf{NLC} derivations of $\xi[A] \Rightarrow C$ and $\phi \Rightarrow B$, and therefore $\xi[\phi \circ B \setminus A] \Rightarrow C$ can be proved therein. \square

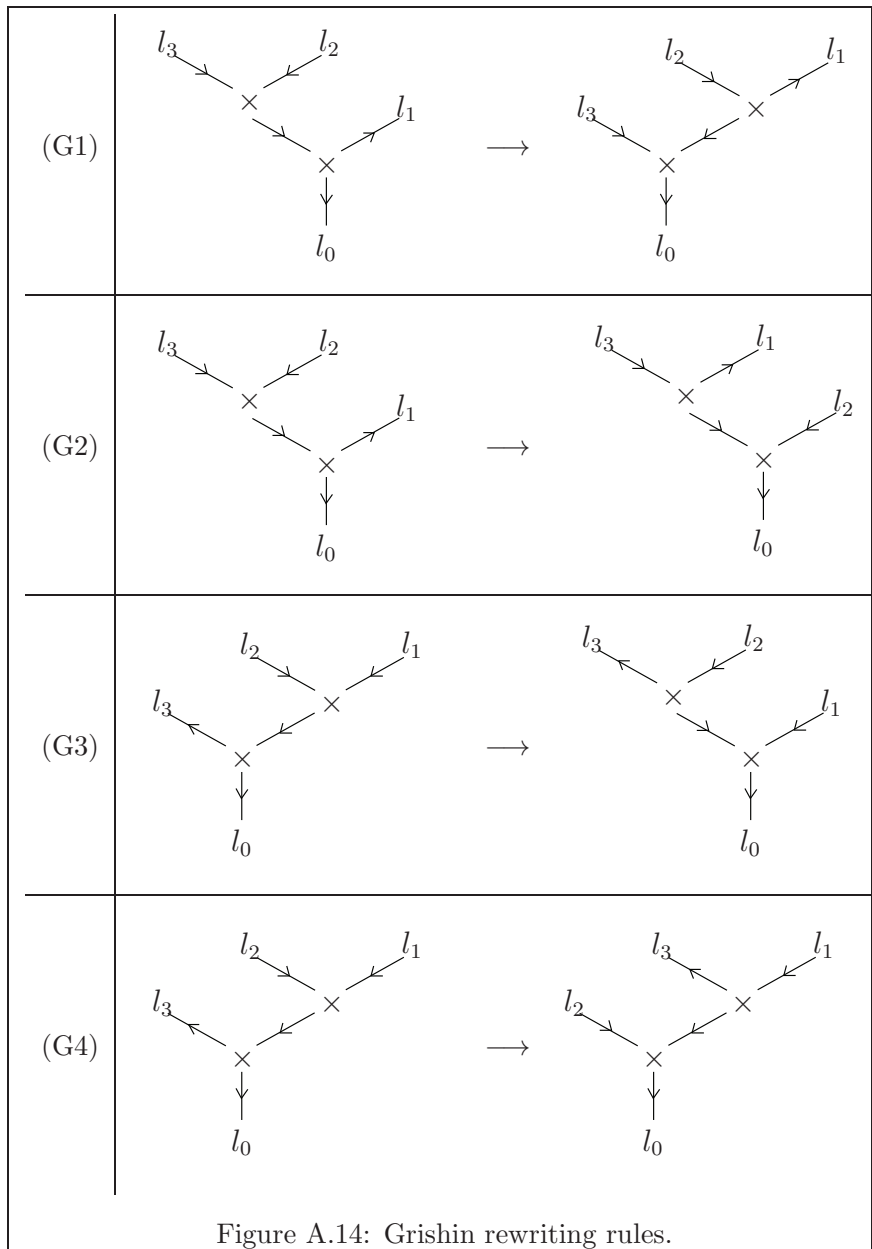
Corollary A.53 *\mathbf{LGC} is a conservative extension of \mathbf{NLC} .*

A.6 Proof net theory for \mathbf{LGC}

The general theory concerning proof nets for the displayed multimodal Lambek Calculus and Theorem A.51 establish the following results, based on a set of rewriting rules that comprise the contraction rules and the Grishin structural rules of Figure A.14.

Proposition A.54 *If \mathcal{D} is an \mathbf{LGC} derivation of a sequent $\sigma \Rightarrow \tau$, then $(\mathcal{D})^-$ rewrites as $\sigma^\bullet \star \tau^\circ$.*

Theorem A.55 (Sequentialization) *Let Π be a \mathbf{LGC}_d proof structure such that Π^- rewrites as $\sigma^\bullet \star \tau^\circ$, where $\sigma \Rightarrow \tau$ is an \mathbf{LGC} sequent. Then there is an \mathbf{LGC} derivation \mathcal{D} of $\sigma \Rightarrow \tau$ such that $(\mathcal{D}) = \Pi$.*



Conclusions

The Lambek Calculus **LC** is an *ante litteram* example of sublinear logic. Roorda's adaptation to it of Girard's proof net theory and the introduction of multimodality opened a new area of investigation: proof nets for multimodal calculi. Moot and Puite's procedural theory of proof nets for Lambek Calculi with weak Sahlqvist structural rules was a brilliant contribution to this field. However, from a declarative point of view, there have been fewer results.

The study –reported in the first part of this thesis– of the invariants for Moot and Puite's rewriting rules along successful derivations resulted in a declarative characterization of the most constrained calculi.

First of all, introducing the notion of *balance* we completed the diamond of proof net theory for the Lambek Calculus with/without Associativity and/or Commutativity.

Our proposal allows three possible approaches to capture sensitivity to structure. At one end of the spectrum, one can verify whether a proof structure is correct with respect to a given structuring of its conclusions. At the other end, one can check correctness of a proof structure *per se* and unearth a possible structuring. An intermediate possibility is to prove correctness of a proof structure with respect to a linear (cyclic) order of the conclusions and recover a structuring compatible with this order.

The latter strategy seems to be more in line with linguistic necessities, since structuring beyond order is a theoretical construct (at least as long as we do not take into account suprasegmental phonology).

Secondly, we have characterized –via the notion of *endomodality*– the lack of interaction among binary modalities. Moreover we have seen that, in endomodal proof structures, sensitivity to order and structure can be checked separately for each modality. This gives an upper limit for the characterization of proof nets, in the sense that any weak Sahlqvist structural rule should be characterized by relaxing endomodality+balance+planarity.

However, this shows also the limits of this approach. For instance, it does not provide a characterization of a multimodal system in which there is communication between two non-associative non-commutative modes, despite the fact that such communication does not affect the geometry of proof structures. Thus the characterization is very fragile. A possible solution would be to find a more geometric criterion. Just as planarity has been reformulated by Maieli in terms of order constraints on trimmings, one could express balance by the following property, *monocentricity*, that I conjecture to be equivalent to balance in any (DR_2) -correct proof structure Π :⁸

for every \wp link L there is a tensor node T_L such that for all trimmings τ at L and for all conclusions C of Π that belong to τ , the node T_L is the center of L and C in τ .

However, as soon as we allow for communication between modalities, it is not clear how to recover the structuring of the conclusions. Hence, a declarative characterization of Multimodal Lambek Calculi is still largely an open question.

The third point addressed in Part I of the thesis is the question of links for unary modalities. Since Danos-Regnier switchings cannot discriminate among unary links for a unary modality and its residuated operator, there arises the necessity of treating unary modalities by compilation.

While working on the last chapter on pregroups with brackets, it became clear that Versmissen's proposal of mimicking unary modalities with two extra symbols was not correct. The proof net theory elaborated for bracketed pregroups suggested also how to fix his proposal. The solution, however, is based on planarity. This is not a good strategy if one wants to use modalities to license structural rules. Hence the proposal for a translation that makes use of only one extra symbol and an extra modality, that does not interact with the previous ones. A proof net theory is obtained immediately, compiling the translation and using the endomodality correctness criterion.

Observe also that if there are several families of unary modalities, the translation can be iterated. However, a certain amount of redundancy is introduced, in that in the translation of any unary modality, the modality is recorded both on the new atomic type that is introduced and on the binary modality used for the translation. There seem to be two strategies to avoid this redundancy:

- assume that for all unary modalities h and k the new atomic types b_h and b_k coincide;

⁸Recall Definition 2.30 of the center of a \wp link L and a conclusion in a trimming at L .

- use different atomic types, but only one new non-associative binary modality.

The latter strategy appears perhaps more conservative and natural. However, if brackets are used to record, say, grammatical features and some communication among features is expected (say, a noun marked for masculine gender is marked for gender), then the former strategy is preferable, since the communication can be encoded as structural rules. But again, as soon as there is intermodal communication we have no general theory for unearthing the structuring of the conclusions.

In Part II of the thesis we have directed our attention to recent theories that have moved away from multimodality and the use of structural rules. We have limited our attention to those theories for which there is research on their proof nets.

Grafting Puite and Moot’s procedural theory on *directed* graphs yields a proof net theory for the Lambek-Grishin Calculus. However, the rewriting rules highlight that this calculus is not truly without structural rules as might appear from the sequent presentation.

In contrast, Morrill’s Discontinuous Hypercalculus is an example of a calculus that is truly monomodal and without structural rules. We have provided it with a theory of proof nets obtained constraining proof nets of the Commutative Lambek Calculus. Moreover, we have characterized those proof nets that can be sequentialized without making use of empty sequents.

One would also like to extend the proof net theory to include the nondeterministic wrap, an operator with a distinctively additive flavour. It seems that one possible solution would be to combine the theory proposed here with the theory proposed by [47] for additives in (commutative) linear logic. The combination is certainly possible for the Lambek Calculus enriched with additives, provided that local correctness is checked with respect to a given order.

But, perhaps, it is wrong to think of the nondeterministic wrap as an additive operator. It should be viewed rather as being some sort of *internally* commutative operator, the commutativity being among the various points of discontinuity. From the point of view of proof nets, it seems that this internal commutativity could be treated following Mellies’s approach, i.e. introducing also switchings for nondeterministic tensor operators. However, it is not clear to me how we could write a non-additive sequent calculus for such an operator. And then, if we could, this would also mean that we are reintroducing multimodality.

The other contribution offered in Part II is concerned with enriching pre-groups with unary modalities. We propose a definition of bracketed pre-group, explain its free construction and generalize the Switching Lemma. We observe that Francez and Kaminski's commutativity and cancelability inequalities can be imposed on this construction unproblematically. A more difficult question, that we leave for further research, is whether their inequations could be stated *using* unary modalities.

In any case, it seems that multimodality keeps lurking on the horizon ... Waiting for further developments in the field, allow me to conclude borrowing some lines from Queneau's *Exercices de style*:

Tiens j'ai déjà raconté la moitié de mon histoire. Je me demande comment j'ai fait. C'est tout de même agréable d'écrire. Mais il reste le plus difficile. Le plus calé. La transition. D'autant plus qu'il n'y a pas de transition. Je préfère m'arrêter.[†]

[†]Hm, I've got through half my story already. Wonder how I did it. Writing's really quite pleasant. But there's still the most difficult part left. The part where you need the most know-how. The transition. All the more so as there isn't any transition. I'd rather stop here. (From the English edition, translated by B. Wright)

Bibliography

- [1] V. M. Abrusci. Phase Semantics and Sequent Calculus for Pure Non-commutative Classical Linear Propositional Logic. *The Journal of Symbolic Logic*, 56(4), 1991.
- [2] V. M. Abrusci and P. Ruet. Non-commutative Logic I: the multiplicative fragment. *Annals of Pure and Applied Logic*, 101(1), 2000.
- [3] A. Avron. A constructive analysis of RM. *Journal of Symbolic Logic*, 52(4):939–951, 1987.
- [4] E. Bach. Discontinuous Constituents in Generalized Categorical Grammars. In *Proceedings of the 11th Annual Meeting of the North Eastern Linguistic Society*, New York, 1981.
- [5] E. Bach. Some Generalizations of Categorical Grammar. In F. Landman and F. Veltman, editors, *Varieties of Formal Semantics*. Fortis, 1984.
- [6] J. Baldridge and Geert-Jan M. Kruijff. Multi-modal combinatory categorical grammar. In *Proceedings of the 11th Conference of EACL*. Budapest, Hungary, 2003.
- [7] D. Bargelli and J. Lambek. An algebraic approach to arabic sentence structure. *Linguistic Analysis*, 31:301–315, 2001.
- [8] D. Bargelli and J. Lambek. An algebraic approach to french sentence structure. In de Groote P. et al., editor, *Logical aspects of computational linguistics*. Springer LNAI, Berlin, 2001.
- [9] D. Bargelli and J. Lambek. An algebraic approach to turkish syntax and morphology. *Linguistic Analysis*, 34:66–84, 2004.
- [10] G. Bellin and J. van de Wiele. Empires and kingdoms in \mathbf{MLL}^- . In Jean-Yves Girard & Yves Lafont & Laurent Regnier, editor, *Advances in Linear Logic*, London Mathematical Society Lecture Note Series 222. Cambridge University Press, Cambridge, 1995.

- [11] R. Bernardi and M. Moortgat. Continuation semantics for symmetric categorial grammar. *Lecture Notes in Computer Science*, 2007.
- [12] J. Bresnan. *The Mental Representation of Grammatical Relations*. MIT Press, 1982.
- [13] J. Bresnan. *Lexical Functional Syntax*. Basil Blackwell, 2001.
- [14] W. Buszkowski. Compatibility of categorial grammar with an associated category system. *Z. math. Logik und Grundlagen der Mathematik*, 1982.
- [15] W. Buszkowski. Completeness results for Lambek calculus. *Z. Math. Logik Grundlag. Math.*, 32:13–28, 1986.
- [16] W. Buszkowski. Lambek Grammars Based on Pregroups. In P. de Groote, G. Morrill, and C. Retoré, editors, *Logical Aspects of Computational Linguistics, Proc. LACL 2001*, pages 95–109. Springer, 2001.
- [17] W. Buszkowski. Cut Elimination for the Lambek Calculus of Adjoints. In *New Perspectives in Logic and Formal Linguistics*. Bulzoni, 2002.
- [18] W. Buszkowski. Type logics and pregroups. *Studia Logica*, 87(2-3):145–169, 2007.
- [19] W. Buszkowski and K. Moroz. Pregroup grammars and context-free grammars. In C. Casadio and J. Lambek, editors, *Computational Algebraic Approaches to Natural Language*, pages 1–21. Polimetrica, 2008.
- [20] D. Caplan and N. Hildebrandt. *Disorders of Syntactic Comprehension*, chapter 4. MIT Press, Cambridge MA, 1998.
- [21] B. Carpenter. The Turing Completeness of Multimodal Categorial Grammars. In J. Gerbrandy, M. Marx, M. de Rijke, and Y. Venema, editors, *JFAK: Essays dedicated to Johan van Benthem on the occasion of his 50th birthday*. Institute for Logic, Language, and Computation, 1999.
- [22] A. Chernilovskaya. Relational completeness of the Lambek-Grishin calculus with unary modalities. In *Symmetric Categorial Grammar Course Notes ESSLLI07*. 2007.
- [23] N. Chomsky. *Aspects of the Theory of Syntax*. MIT Press, Cambridge, MA, 1965.
- [24] N. Chomsky. *The Minimalist Program*. MIT Press, 1996.

- [25] V. Danos and L. Regnier. The structure of multiplicatives. *Archive for Mathematical Logic*, 28:181–203, 1989.
- [26] P. de Groote. Partially commutative linear logic: sequent calculus and phase semantics. In *Proofs and Linguistic categories - Applications of Logic to the Analysis and Implementation of Natural Language*, Bologna, 1996. CLUEB.
- [27] P. de Groote. Towards abstract categorial grammars. In *ACL '01: Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 252–259, New York, 2001.
- [28] P. de Groote and F. Lamarche. Classical Non-Associative Lambek Calculus. *Studia Logica*, 71:355–388, 2002.
- [29] P. de Groote, S. Pogodalla, and C. Pollard. About Parallel and Syntactocentric Formalisms: What the Encoding of Convergent Grammar into Abstract Categorial Grammar Tells Us. To appear in *Fundamenta Informaticae*, 2009.
- [30] P. de Groote and Ch. Retoré. On the semantic readings of proof nets. In G-J. M. Kruijff, G. Morrill, and R. T. Oehrle, editors, *Proceedings of Formal Grammar*, pages 57–70, Prague, 1996.
- [31] P. de Groote and Ch. Retoré. Proof-theoretic methods in computational linguistics. In *Lecture Notes 15th European Summer School in Logic, Language and Information*, Vienna, 2003.
- [32] Ph. de Groote and S. Pogodalla. Abstract Categorial Grammars. ESS-LLI Notes, Edinburgh, Scotland, 2005.
- [33] R. Diestel. *Graph Theory*. Springer-Verlag, 2005.
- [34] K. Došen. A brief survey of frames for the Lambek calculus. *Z. Math. Logik Grundlag. Math.*, 38:179–187, 1992.
- [35] M. Fadda. Toward Flexible Pregroup Grammars. In V.M. Abrusci and C. Casadio, editors, *New Perspectives in Logic and Formal Linguistics*, Roma, 2002. Bulzoni. Proceedings Vth Roma Workshop.
- [36] M. Fadda. Non-associativity and balanced proof nets. In *Categorial grammars. An efficient tool for Natural Language Processing*, 2004.
- [37] M. Fadda and G. Morrill. The Lambek Calculus with brackets. In Ph. Scott, C. Casadio, and R. Seely, editors, *Language and Grammar: Studies in Mathematical Linguistics and Natural Language*. CSLI, 2005.

- [38] N. Francez and M. Kaminski. Commutation-Augmented Pregroup Grammars and Mildly Context-Sensitive Languages. *Studia Logica*, 87(2):295–321, 2007.
- [39] G. Frege. Über Sinn und Bedeutung. *Zeitschrift für Philosophie und philosophische Kritik*, 100:25–50, 1892.
- [40] E. Gibson. Linguistic complexity: locality of syntactic dependencies. *Cognition*, 68:1–76, 1998.
- [41] J.-Y. Girard. Quantifiers in Linear Logic II. In G. Corsi and G. Sambin, editors, *Nuovi problemi della logica e della filosofia della scienza, Volume II*. CLUEB, Bologna(Italy), 1991. Proceedings of the conference with the same name, Viareggio, 8-13 gennaio 1990.
- [42] Y. J Girard. Linear Logic. *Theoretical Computer Science*, 50, 1987.
- [43] Goré, R. P. and R. Bernardi. Display Calculi meet Categorical Type Logics. *ESSLLI*, 2004.
- [44] V.N. Grishin. On a generalization of the Ajdukiewicz-Lambek system. In *Studies in Non-classical Logics and Formal Systems*. Nauka, 1983.
- [45] M. Hepple. *The grammar and processing of order and dependency*. PhD thesis, Edinburgh, 1990.
- [46] R. Huddleston. *Introduction to the Grammar of English*. Cambridge Textbooks in Linguistics. Cambridge University Press, Cambridge, 1984.
- [47] D. Hughes and R. van Glabbeek. Proof Nets for Unit-free Multiplicative-Additive Linear Logic. In *Proceedings of the 18th Annual IEEE Symposium on Logic in Computer Science*, 2003.
- [48] M.E. Johnson. Proof nets and the complexity of processing center-embedded constructions. *Journal of Logic, Language, and Information*, 4(7), 1998.
- [49] A. K. Joshi, S. Kulick, and N. Kurtonina. Partial Proof Trees, Hybrid Logic, and Quantifier Scope. *Research on Language and Computation*, 2(1), 2004.
- [50] M. Kanazawa. The Lambek calculus enriched with additional connectives. *JOLLI*, 1992.
- [51] A. Kiślak-Malinowska. On the Logic of β -pregroups. *Studia Logica*, 87:323–342, 2007.

- [52] M. Kołowska-Gawiejnowicz. Powerset Residuated Algebras and Generalized Lambek Calculus. *Math. Log. Quart.*, 43:60–72, 1997.
- [53] N. Kurtonina. *Frames and labels. A modal analysis of categorial inference*. PhD Dissertation OTS Utrecht. ILLC, Amsterdam, 1995.
- [54] N. Kurtonina and M. Moortgat. Structural control. In Blackburn P. and M. de Rijke, editors, *Logic, Structure and Syntax*. Kluwer, 1995.
- [55] N. Kurtonina and M. Moortgat. Relational semantics for the Lambek-Grishin calculus. In *10th Conference on Mathematics of Language*. 2007.
- [56] F. Lamarche and Ch. Retoré. Proof Nets for the Lambek Calculus - an Overview. In V.M. Abrusci and C. Casadio, editors, *Proofs and Linguistic categories - Applications of Logic to the Analysis and Implementation of Natural Language*. Clueb, 1996.
- [57] J. Lambek. The mathematics of Sentence Structure. *American Math. Monthly*, 65:154–170, 1958.
- [58] J. Lambek. On the calculus of syntactic types. In Roman Jakobson, editor, *Structure of language and its mathematical aspects, Proc. of the Symposia in Applied Mathematics XII*. American Mathematical Society, 1961.
- [59] J. Lambek. From categorial grammar to bilinear logic. In K. Došen and P. Schroeder-Heister, editors, *Substructural Logics*. Oxford University Press, 1993.
- [60] J. Lambek. Type grammars revisited. In Lecomte A., Lamarche F., and G. Perrier, editors, *Logical Aspects of Computational Linguistics*. Springer, 1997.
- [61] J. Lambek. Pregroups: a new algebraic approach to sentence structure. In C. Martin-Vide and G. Paun, editors, *Recent Topics in Mathematical and Computational Linguistics*. Editura Academici Romne, 2000.
- [62] J. Lambek. *From word to sentence: a computational algebraic approach to grammar*. Polimetrica, Milano, 2008.
- [63] Joachim Lambek. Should Pregroup Grammars be Adorned with Additional Operations? *Studia Logica*, 87(2-3):343–358, 2007.
- [64] A. Lecomte. Categorial grammar for minimalism. *Language and Grammar : Studies in Mathematical Linguistics and Natural Language*, CSLI Lecture Notes(168):163–188, 2005.

- [65] R. Maieli. A new correctness criterion for multiplicative non- commutative proof-nets. *Archive for Mathematical Logic*, 42, 2003.
- [66] M. Melissen. The Generative Capacity of the Lambek–Grishin Calculus: A New Lower Bound. In P. de Groote, editor, *Proceedings of the 14th Formal Grammar Conference*, volume 5591 of *Lecture Notes in Computer Science*. Springer, 2009. to appear.
- [67] P.A. Melliès. A topological correctness criterion for non-commutative logic. In Ehrhard T., J-Y. Girard, P. Ruet, and P. Scott, editors, *Linear Logic in Computer Science*. Cambridge University Press, 2004.
- [68] M. Moortgat. *Categorical Investigations: Logical and Linguistic Aspects of the Lambek Calculus*. Foris, Dordrecht, 1988.
- [69] M. Moortgat. Categorical type logics. In J. van Benthem and V. ter Meulen, editors, *Handbook of logic and language*. Elsevier Science B.V., 1997.
- [70] M. Moortgat. Symmetries in Natural Language Syntax and Semantics: The Lambek-Grishin Calculus. In *Logic, Language, Information and Computation*, volume 4576, pages 264–284. Springer-Verlag, 2007.
- [71] M. Moortgat and G. Morrill. Heads and Phrases. Type Calculus for Dependency and Constituent Structure. Manuscript, OTS, 1991.
- [72] M. Moortgat and D. Oehrle. Pregroups and type-logical grammar: searching for convergence. In C. Casadio, P. Scott, and R.A.G. Seely, editors, *Language and grammar: Studies in mathematical linguistics and natural language*. CSLI, 2004.
- [73] Michael Moortgat. A context calculus for LG.
- [74] R. Moot. *Proof Nets for Linguistic Analysis*. PhD thesis, Utrecht, 2002.
- [75] R. Moot. Proof nets for display logic. *Technical report*, 2007. <http://arxiv.org/abs/0711.2444>.
- [76] R. Moot and M. Piazza. Linguistic applications of first order intuitionistic linear logic. *Journal of Logic, Lanugage and Information*, 10:211–232, 2001.
- [77] R. Moot and Q. Puite. Proof nets for the multimodal Lambek Calculus. *Studia Logica*, 71(3), 2002.
- [78] G. Morrill. Discontinuity and Pied Piping in Categorical Grammar. Universitat Politècnica de Catalunya, 1993. Research Report, LSI-93-18-R.

- [79] G. Morrill. *Type Logical Grammar: Categorical Logic of Signs*. Kluwer, 1994.
- [80] G. Morrill. Relational interpretation and geometrical form. In V.M. Abrusci and C. Casadio, editors, *Dynamic Perspectives in Logic and Linguistics*, pages 145–182. Bulzoni, 1999.
- [81] G. Morrill. Incremental processing and acceptability. *Computational Linguistics*, 26(3):319–338, 2000.
- [82] G. Morrill. *Categorical Grammar Logical Syntax, Semantics, and Processing*. Claredon Press, Oxford, 2010. (forthcoming).
- [83] G. Morrill and M. Fadda. Proof Nets for Basic Discontinuous Lambek Calculus. *Journal of Logic and Computation*, 18:239–256, 2008.
- [84] G. Morrill, M. Fadda, and O. Valentín. Nondeterministic Discontinuous Lambek Calculus. In J. Geertzen, E. Thijsse, H. Bunt, and A. Schiffrin, editors, *Proceedings of the Seventh International Workshop on Computational Semantics, IWCS-7*, page 129141, 2007.
- [85] G. Morrill and A. Gavarró. On Aphasic Comprehension and Working Memory Load. In *Categorical grammars. An efficient tool for Natural Language Processing*, 2004.
- [86] G. Morrill and J.M. Merenciano. Generalising discontinuity. *Traitement automatique des langues*, 37(2):119–143, 1996.
- [87] G. Morrill and O. Valentín. Displacement Calculus. To appear in the Lambek Festschrift, special issue of *Linguistic Analysis*. Available at <http://arxiv.org/abs/1004.4181>, 2010.
- [88] G. Morrill and O. Valentín. On Anaphora and the Binding Principles in Categorical Grammar. In *Proceedings of WoLLIC 2010, 17th Workshop on Logic, Language, Information and Computation*, 2010.
- [89] G. Morrill and O. Valentín. On Calculus of Displacement. In *Proceedings of TAG+Related Formalisms*. University of Yale, 2010.
- [90] G. Morrill, O. Valentín, and M. Fadda. Discontinuous Lambek Calculus. *Manuscript*, 2008.
- [91] G. Morrill, O. Valentín, and M. Fadda. Dutch Grammar and Processing: A Case Study in TLG. In P. Bosch, D. Gabelaia, and J. Lang, editors, *Logic, Language, and Computation: 7th International Tbilisi Symposium, Revised Selected Papers*, pages 272–286, 2009.
- [92] R. Muskens. Categorical grammar and discourse representation theory. In *Proceedings COLING 94*, pages 508–514, Kyoto, 1994.

- [93] R. Muskens. Categorical Grammar and Lexical-Functional Grammar. In *Proceedings of the LFG01 Conference*, pages 259–279, University of Hong Kong, Hong Kong, 2001. CSLI.
- [94] R.T. Oehrle. Multimodal type logical grammar. In R. Borsley and K. Borjars, editors, *Non-Transformational Syntax*. Blackwell, to appear.
- [95] B.H. Partee and H.L.W. Hendriks. Montague grammar. In J. van Benthem and A. ter Meulen, editors, *Handbook of logic and language*. Elseviere Science B.V., 1997.
- [96] M. Pentus. Lambek grammars are context-free. Technical report, Dept.Math.Logic, Steklov Math. Institute, Moskow, 1992.
- [97] M. Pentus. Models for the Lambek calculus. *Annals of Pure and Applied Logic*, 75:1-2:179–213, 1995.
- [98] M. Pentus. Free monoid completeness of Lambek calculus allowing empty premises. In J.M. & Lascar D. & Mints G. Larrazabal, editor, *Logic colloquium '96*, volume 12, pages 171–209. Springer, 1998. Lecture notes in Logic.
- [99] M. Pentus. Lambek calculus is NP-complete. Technical report, Cuny University–Program in Computer Science, 2003.
- [100] Mati Pentus. Product-free lambek calculus and context-free grammars. *The Journal of Symbolic Logic*, 62:648–660, 1997.
- [101] F. Pfenning. Structural cut elimination. In *Proceedings of the Tenth Annual Symposium on Logic in Computer Science*, San Diego, 1995. IEEE Computer Society Press.
- [102] C. Pollard. *Generalized Phrase Structure Grammars, Head Grammars, and Natural Language*. PhD thesis, Stanford University, 1984.
- [103] C. Pollard. An Introduction to Convergent Grammar. Presented at the Sminaire Calligramme. Nancy, France, June 2008. Available at <http://www.ling.ohio-state.edu/pollard/cvg/calliho.pdf>, 2008.
- [104] C. Pollard and I. A. Sag. *Information-based Syntax and Semantics*. CSLI, Standford, CA, 1987.
- [105] C. Pollard and I. A. Sag. *Head-driven Phrase Structure Grammar*. CSLI, Standford, CA, 1994.
- [106] C. Pollard and T. Wasow. *Syntactic Theory: A Formal Introduction*. CSLI, 1999.

- [107] G. Pottinger. Uniform, cut-free formulations of T, S4 and S5. *Journal of Symbolic Logic*, 48(3), 1983.
- [108] Q. Puite. *Sequents and Link Graphs*. PhD thesis, Utrecht, 2001.
- [109] Ch. Retoré. Calcul de Lambek et Logique Linéaire. *Traitement automatique du langage*, 2(37):39–70, 1995.
- [110] D. Roorda. *Resource Logics. Proof-theoretical Investigations*. PhD thesis, Universiteit van Amsterdam, 1991.
- [111] J.R. Ross. *Constraints on variables in syntax*. PhD thesis, MIT, 1967.
- [112] P. Ruet. Non-commutative Logic II: sequent calculus and phase semantics. *Mathematical Structures in Computer Science*, 10(2):277–312, 2000.
- [113] I.A. Sag. On parasitic gaps. *Linguistics and Philosophy*, 6(1):35–45, 1983.
- [114] Y. Savateev. Product-Free Lambek Calculus Is NP-Complete. In *Logical Foundations of Computer Science. Proceedings of the 2009 International Symposium on Logical Foundations of Computer Science*, pages 380–394, Berlin, 2009. Springer-Verlag.
- [115] Solà, J. and M.R. Lloret and J. Mascaró and M. Pérez Saldanya, editor. *Gramàtica del català contemporani*. Empúries, Barcelona, 2002.
- [116] E. Stabler. Tupled pregroup grammars. Technical report, UCLA, 2003.
- [117] E. P. Stabler. Derivational minimalism. In C. Retoré, editor, *LACL '96: Selected papers from the First International Conference on Logical Aspects of Computational Linguistics*, pages 68–95, London, UK, 1997. Springer-Verlag.
- [118] A. Teodori. Applicazioni alla linguistica delle reti di dimostrazione della logica lineare. La Sapienza, Roma, 2003. Tesi di Laurea.
- [119] O. Valentin. The 1-Discontinuous Lambek Calculus: Type Logical Grammar and discontinuity in natural language, 2006. Available at <http://seneca.uab.es/ggt/publicacions/tesis/pdf/OValentin.pdf>.
- [120] J. van Benthem. The semantic of variety in categorial grammar. In W. Marciszewski Buszkowski, W. and J. van Benthem, editors, *Categorial grammar*. Benjamin, Amsterdam, 1988.
- [121] J. van Benthem. *Language in action. Categories, Lambdas, and Dynamic Logic*. Studies in Logic. North-Holland, Amsterdam, 1991.

- [122] K. Versmissen. *Grammatical composition: modes, models and modalities*. PhD thesis, OTS Utrecht, 1996.
- [123] D. N. Yetter. Quantales and (noncommutative) linear logic. *Journal of Symbolic Logic*, 55(1):41–64, 1990.