

Categorial Type Logics

Michael Moortgat

Utrecht Institute of Linguistics, OTS

Excerpt from Chapter Two in J. van Benthem and A. ter Meulen (eds.)

Handbook of Logic and Language. Elsevier, 1997.

Contents

1	Introduction: grammatical reasoning	1
2	Linguistic inference: the Lambek systems	5
2.1	Modeling grammatical composition	5
2.2	Gentzen calculus, cut elimination and decidability	9
2.3	Discussion: options for resource management	13
3	The syntax-semantics interface: proofs and readings	16
3.1	Term assignment for categorial deductions	17
3.2	Natural language interpretation: the deductive view	21
4	Grammatical composition: multimodal systems	26
4.1	Mixed inference: the modes of composition	26
4.2	Grammatical composition: unary operations	30
4.2.1	Unary connectives: logic and structure	31
4.2.2	Applications: imposing constraints, structural relaxation	35
4.2.3	Resource control: faithful embeddings	38
5	Reasoning about multiple type assignments	40
6	Hybrid architectures	40
7	Categorial parsing as deduction	41
8	Conclusions, directions for further research	41

Definitions

- (5) Binary multiplicatives: formula language.
- (2.1) Binary multiplicatives: frame semantics.
- (2.2) The pure residuation logic **NL**.
- (2.4) Canonical model.
- (2.5) Resource management postulates.
- (2.7) Weak Sahlqvist axioms.
- (6) Groupoid interpretation.
- (2.9) Combinator proof terms.
- (2.12) Gentzen calculus: **NL**.
- (8) Gentzen calculus: structural rules.
- (2.13) Implicit structural rules.
- (9) The Cut rule.
- (2.17) Natural Deduction.
- (11) Lifting as closure operation.
- (13) Semantic domains.
- (3.1) Syntax of typed λ terms.
- (3.2) Term assignment: **LP**.
- (3.3) **LP** proof terms.
- (3.5) Term equations and their proof-theoretic reflexes.
- (14) Category-to-type mapping.
- (3.8) Term assignment: the sublinear case.
- (3.9) Term assignment: Natural Deduction.
- (17) Argument Raising.
- (3.14) Binding 'in situ'.
- (19) Binary multiplicatives: multimodal vocabulary.
- (4.1) Interpretation: multimodal frames.
- (20) Residuation: mode restrictions.
- (4.2) Multimodal Gentzen calculus.
- (4.3) Interaction principles: Mixed Associativity/Commutativity.
- (22) Interaction postulate: Wrapping.
- (4.6) Restricted Contraction.
- (24) Generalized residuation.
- (4.8) Unary multiplicatives \diamond, \square : interpretation, residuation laws.
- (4.9) Unary multiplicatives: combinator presentations.
- (27) Unary multiplicatives: logical and structural connectives.
- (4.10) Unary multiplicatives: Gentzen rules.
- (4.11) Unary multiplicatives: structural options.
- (4.12) $S4$: sugared Gentzen presentation.
- (4.13) Simulating T and 4 via compilation $\diamond\square$.
- (4.14) Modal Commutativity, Modal Associativity.
- (4.16) Unary multiplicatives: term assignment.
- (4.17) Unary multiplicatives: term equations.
- (28) $S4$ modalities: intensional semantics.
- (4.20) Embedding translations: **NL** versus **L**.
- (4.22) Associativity: global versus modal.

Propositions

- (2.3) Completeness: **NL**.
- (2.6) Completeness: **L, NLP, LP**.
- (2.8) Sahlqvist completeness.
- (2.11) Equivalence of axiomatic and Gentzen presentations.
- (2.14) Cut Elimination.
- (2.18) Characteristic theorems, derived rules of inference.
- (3.4) Correspondence: proofs and terms.
- (4.21) Dominance: recovering control.
- (4.23) Dominance: licensing structural relaxation.

Examples

- (2.10) Combinators: application, lifting.
- (2.15) Cut Elimination: permutation conversion.
- (2.16) Cut Elimination: detour conversion.
- (2.19) Deriving Associativity from Geach.
- (2.20) Restructuring: S(VO) versus (SV)O.
- (2.21) Non-constituent coordination.
- (12) Coordinate Structure Constraint violations.
- (2.22) Mixed Composition.
- (3.6) Principal Cut: β conversion.
- (3.7) Pruning complex axioms: η conversion.
- (3.10) Argument lowering: $(np \setminus s) / ((s / np) \setminus s) \rightarrow (np \setminus s) / np$.
- (3.11) Argument raising: $(np \setminus s) / np \rightarrow ((s / (np \setminus s)) \setminus s) / ((s / np) \setminus s)$.
- (3.12) Non-linear lexical semantics: 'Everyone loves himself'.
- (15) DRT lexical type assignments.
- (3.13) Type-driven composition of Discourse Representation Structures.
- (16) Scoping: peripheral versus medial cases.
- (18) Flexible Montague Grammar: type shifting.
- (3.15) 'In situ' binding.
- (4.4) Mixed Composition as a multimodal theorem.
- (4.5) Multimodal definition of 'in situ' binding.
- (23) Combinator **S** and parasitic gaps.
- (4.7) Combinator **S** as a multimodal theorem.
- (29) Licensing structural relaxation: controlled Associativity, Commutativity.
- (30) Controlled restructuring: $\diamond \square$ compilation.
- (4.18) Island constraints.
- (31) Word-order domains.
- (32) Reflexives: clause-boundedness.
- (4.19) Blocking locality violations via $S4 \square$ decoration.
- (4.24) Associativity: imposing constraints versus structural relaxation.

Other Handbook Chapters

- | | |
|----------------|--|
| CHAPTER ONE | Montague Grammar (Partee) |
| CHAPTER THREE | Discourse Representation Theory (Kamp/van Eijck) |
| CHAPTER SEVEN | Compositionality (Janssen) |
| CHAPTER EIGHT | Feature Logics (Rounds) |
| CHAPTER TWELVE | Mathematical Linguistics and Proof Theory (Buszkowski) |

Les quantités du langage et leurs rapports sont régulièrement exprimables dans leur nature fondamentale, par des formules mathématiques. ... L'expression simple [of linguistic concepts] sera algébrique ou elle ne sera pas. ... On aboutit à des théorèmes qu'il faut démontrer.

Ferdinand de Saussure¹

This chapter describes the framework of categorial type logic, a grammar architecture that can be seen as the logical development of the categorial approach to natural language analysis initiated in the Thirties in the work of Ajdukiewicz ([Ajdukiewicz 35]). The choice of grammatical number in the title stresses the pluralistic nature of the enterprise: this chapter systematically charts a *landscape* of systems of grammatical inference — the categorial counterpart of the Chomsky Hierarchy in the framework of phrase structure grammars.

The chapter has been written for two types of readers. For the reader with a background in *linguistics*, it tries to provide a useful compendium of the logical tools and results one needs to appreciate current categorial research. Such a compendium, we hope, will make the research literature more accessible. The reader with a *logic* background is justified in classifying the grammar formalism discussed here under the rubric Applied Logic. To measure progress in this field, then, one has to be in a position to evaluate the ‘closeness of fit’ between the formal systems proposed and the linguistic reality they intend to model. For the logical part of the audience, we try to provide enough linguistic background to make it possible to assess the motivation for categorial design choices.

In organizing the material one can opt for a ‘historical’ mode of development, or for a state-of-the-art presentation of the ‘internal dynamics’ of the field. Only the second approach adequately reveals the connections between linguistic composition and logical deduction. The major organizing principle for the chapter is the vocabulary of the ‘logical constants’ of grammatical reasoning, the type-forming operators. A brief preview is given below.

MULTIPLICATIVE OPERATORS: $\diamond, \square, /, \bullet, \setminus$. The core part of the vocabulary. Unary and binary connectives dealing with grammatical composition in the form and meaning dimensions.

BOOLEAN AND/OR ADDITIVE OPERATORS: $\wedge, \vee, \sqcap, \sqcup$. Conjunctive/disjunctive type specifications, with set-theoretic or additive interpretation in the sense of Linear Logic.

POLYMORPHIC TYPES: first and second order quantification $\forall^1, \exists^1, \forall^2, \exists^2$. Dependent types and type schemata expressing generalizations accross type assignments.

Given the space limitations of the handbook format, the information packaging of this chapter will be dense. Fortunately, we can refer the interested reader to a number of monographs that deal with logical and linguistic aspects of the type-logical approach in a more spacious manner. For the general logical background, *Language in Action* ([van Benthem 91,95]) is essential reading. *Type Logical Grammar* ([Morrill 94a]) situates the type-logical approach within the framework of Montague’s Universal Grammar and presents detailed linguistic analyses for a substantive fragment of syntactic and semantic phenomena in the grammar of English. *Type Logical Semantics* ([Carpenter 96]) offers a general introduction to natural language semantics studied from a type-logical perspective. Chapter Twelve of this handbook discusses categorial grammar logics from the perspective of mathematical linguistics and logical proof theory.

1 Introduction: grammatical reasoning

The central objective of the type-logical approach is to develop a uniform *deductive* account of the composition of form and meaning in natural language: formal grammar is presented as a *logic* — a

¹N10 and N13a in R. Godel *Les sources manuscrites du CLG de F. de Saussure*, Genève, 1957. Quoted without reference by Roman Jakobson in his introduction of [Jakobson 61].

system for reasoning about structured linguistic resources. In the sections that follow, the model-theoretic and proof-theoretic aspects of this program will be executed in technical detail. First, we introduce the central concept of ‘grammatical composition’ in an informal way. It will be useful to distinguish two aspects of the composition relation: a fixed *logical* component, and a variable *structural* component. We discuss these in turn.

GRAMMATICAL COMPOSITION: LOGIC. The categorial perspective on the form-meaning articulation in natural language is based on a distinction, which can be traced back to Frege, between ‘complete’ and ‘incomplete’ expressions. Such a distinction makes it possible to drastically simplify the traditional Aristotelian theory of categories (or types): one can reserve atomic category names for the complete expressions, and for the categorization of the incomplete expressions one inductively defines an infinite supply of category names out of the atomic types and a small number of type-forming connectives.

For the categorization of incomplete expressions, Ajdukiewicz in his seminal [Ajdukiewicz 35] used a fractional notation $\frac{A}{B}$, inspired by Husserl’s *Bedeutungskategorien* and Russell’s *Theory of Types*. The fractional notation immediately suggests the basic combination schema via an analogy with multiplication: $\frac{A}{B} \times B$ yields A . Bar-Hillel (see the papers in [Bar-Hillel 64]) refined the impractical fractional notation by splitting up $\frac{A}{B}$ into a division from the left $B \setminus A$ and a division from the right A/B , in order to discriminate between incomplete expressions that will produce an expression of type A when composed with an arbitrary expression of type B to the left, and to the right, respectively.

It will be helpful for what follows to take a logical (rather than arithmetical) perspective on the category formulas, and read A/B , $B \setminus A$ as directionally-sensitive ‘implications’ — implications with respect to structural composition of linguistic material, rather than logical conjunction of propositions. Let us write $\Gamma \vdash A$ for the basic judgement of the grammar logic: the judgement that the structured configuration of linguistic expressions Γ can be categorized as a well-formed expression of type A . The inference pattern (1) tells us how to arrive at a grammaticality judgement for the composite structure Γ, Δ from judgements for the parts Γ and Δ — it tells us how we can *use* the implications $/$ and \setminus in grammatical reasoning. Where the premises are immediate, the basic law of grammatical composition takes the form of a Modus Ponens inference: $A/B, B \vdash A$ and $B, B \setminus A \vdash A$.

$$\text{from } \Gamma \vdash A/B \text{ and } \Delta \vdash B, \text{ infer } \Gamma, \Delta \vdash A \quad \text{from } \Gamma \vdash B \text{ and } \Delta \vdash B \setminus A, \text{ infer } \Gamma, \Delta \vdash A \quad (1)$$

In the example (2), one finds a little piece of grammatical reasoning leading from lexical categorizations to the conclusion that ‘Kazimierz talks to the mathematician’ is a well-formed sentence. In this example, sentences s , (proper) noun phrases np , common nouns n , and prepositional phrases pp are taken to be ‘complete expressions’, whereas the verb ‘talk’, the determiner ‘the’ and the preposition ‘to’ are categorized as incomplete with respect to these complete phrases. The sequence of Modus Ponens inference steps is displayed in the so-called Natural Deduction format, with labels $/E$, $\setminus E$ for the ‘elimination’ of the implication connectives.

$$\begin{array}{c}
 \text{the} \quad \text{mathematician} \\
 \hline
 \text{to} \quad \frac{(np/n) \quad n}{/E} \\
 \hline
 \text{talks} \quad \frac{(pp/np) \quad np}{/E} \\
 \hline
 \text{Kazimierz} \quad \frac{((np \setminus s)/pp) \quad pp}{/E} \\
 \hline
 \frac{np \quad (np \setminus s)}{\setminus E} \\
 s
 \end{array} \quad (2)$$

The inferences of (1) build more complex structural configurations out of their parts by using the grammatical implications. What about looking at grammatical structure from the opposite perspective? In other words: given information about the categorization of a composite structure,

GRAMMATICAL COMPOSITION: STRUCTURE. An aspect we have ignored so far in our discussion of the Modus Ponens and Hypothetical Reasoning inferences is the *management* of the linguistic resources — the manipulations we allow ourselves in using linguistic assumptions. Some aspects of resource management are explicitly encoded in the logical vocabulary — the distinction between the ‘implications’ / and \, for example, captures the fact that grammatical inference is sensitive to the linear order of the resources. But other equally important aspects of resource management have remained implicit. In the relative clause example, we inferred $\Gamma \vdash A/B$ from $\Gamma, B \vdash A$. In withdrawing the hypothetical B assumption, we didn’t take into account the *hierarchical embedding* of the B resource: we ignored its vertical nesting in the configuration of assumptions. Resource management, in other words, was implicitly taken to be *associative*: different hierarchical groupings over the same linear ordering of assumptions were considered as indistinguishable for the purposes of grammatical inference. On closer inspection, the implicit claim that restructuring of resources would not affect derivability (grammatical well-formedness) might be justified in some cases, whereas in other cases a more fine-grained notion of grammatical consequence might be appropriate. Similarly, the sensitivity to linear order, which restricts hypothetical reasoning to the withdrawal of a *peripheral* assumption, might be too strong in some cases. Compare (4) with the variant ‘whom Kazimierz talked to yesterday’, where one would like to withdraw a hypothetical np assumption from the non-peripheral position ‘Kazimierz talked to np yesterday’. Switching to a commutative resource management regime would be too drastic — we would not be able anymore to deductively distinguish between the well-formed relative clause and its ill-formed permutations. In cases like this, one would like the grammar logic to provide facilities for *controlled modulation* of the management of linguistic resources, rather than to implement this in a global fashion as a hard-wired component of the type-forming connectives.

A BRIEF HISTORY OF TYPES. The above discussion recapitulates the crucial phases in the historical development of the field. The Modus Ponens type of reasoning, with its functional application interpretation, provided the original motivation for the development of categorial grammar in [Ajdukiewicz 35]. The insight that Modus Ponens and Hypothetical Reasoning are two inseparable aspects of the interpretation of the ‘logical constants’ /, \ is the key contribution of Lambek’s work in the late Fifties. In the papers [Lambek 58, Lambek 61] it is shown that attempts to generalize Modus Ponens in terms of extra rule schemata, such as Type Lifting or Functional Composition, are in fact weak approximations of Hypothetical Reasoning: viewing the type-forming operators as logical connectives, such schemata are reduced to *theorems*, given appropriate resource management choices. In retrospect, one can see that the core components of the type-logical architecture were worked out in 1958. But it took a quarter of a century before Lambek’s work had a clear impact on the linguistic community. Contributions such as [Lyons 68, Lewis 72, Geach 72] are continuations of the rule-based Ajdukiewicz/Bar-Hillel tradition. Ironically, when linguists developed a renewed interest in categorial grammar in the early Eighties, they did not adopt Lambek’s deductive view on grammatical composition, but fell back on an essentially rule-based approach. The framework of Combinatory Categorial Grammar (CCG, [Steedman 88]) epitomizes the rule-based generalized categorial architecture. In this framework, laws of type change and type combination are presented as theoretical primitives (‘combinators’) as a matter of methodological principle. For a good tutorial introduction to CCG, and a comparison with the deductive approach, we refer the reader to [Steedman 93].

The 1985 Tucson conference on Categorial Grammar brought together the adherents of the rule-based and the deductive traditions. In the proceedings of that conference, *Categorial Grammars and Natural Language Structures* ([Oehrle e.a. 88]) one finds a comprehensive picture of the varieties of categorial research in the Eighties.

Lambek originally presented his type logic as a calculus of *syntactic* types. Semantic interpretation of categorial deductions along the lines of the Curry-Howard correspondence was put on the categorial agenda in [van Benthem 83]. This contribution made it clear how the categorial type logics realize Montague’s Universal Grammar program — in fact, how they improve on Montague’s own execution of that program in offering an integrated account of the composition of linguistic

meaning *and* form. Montague’s adoption of a categorial syntax does not go far beyond notation: he was not interested in offering a principled theory of allowable ‘syntactic operations’ going with the category formalism.

The introduction of Linear Logic in [Girard 87] created a wave of research in the general landscape of ‘substructural’ logics: logics where structural rules of resource management are controlled rather than globally available. The importance of the distinction between the logical and the structural aspects of grammatical composition is a theme that directly derives from this research. The analysis of the linguistic ramifications of this distinction has guided the development of the present-day ‘multimodal’ type-logical architecture to be discussed in the pages that follow.

2 Linguistic inference: the Lambek systems

In the following sections we present the basic model-theory and proof-theory for the logical constants $/, \bullet, \backslash$, the so-called *multiplicative* connectives. On the model-theoretic level, we introduce abstract mathematical structures that capture the relevant aspects of grammatical composition. On the proof-theoretic level, we want to know how to perform valid inferences on the basis of the interpreted type language. We are not interested in syntax as the manipulation of meaningless symbols: we want the grammatical proof-theory to be sound and complete with respect to the abstract models of grammatical composition.

We proceed in two stages. In the present section, we develop a landscape of *simple* Lambek systems. Simple Lambek systems are obtained by taking the logic of residuation for a family of multiplicative connectives $/, \bullet, \backslash$, together with a package of structural postulates characterizing the resource management properties of the \bullet connective. As resource management options, we consider Associativity and Commutativity. Different choices for these options yield the type logics known as **NL**, **L**, **NLP**, **LP**. Each of these systems has its virtues in linguistic analysis. But none of them in isolation provides a basis for a realistic theory of grammar. Mixed architectures, which overcome the limitations of the simple systems, are the subject of §4.

2.1 Modeling grammatical composition

Consider the language \mathcal{F} of category formulae of a simple Lambek system. \mathcal{F} is obtained by closing a set \mathcal{A} of atomic formulae (or: basic types, prime formulae, e.g. s, np, n, \dots) under binary connectives (or: type forming operators) $/, \bullet, \backslash$. We have already seen the connectives $/, \backslash$ at work in our informal introduction. The \bullet connective will make it possible to explicitly refer to composed structures — in the introduction we informally used a comma for this purpose.

$$\mathcal{F} ::= \mathcal{A} \mid \mathcal{F}/\mathcal{F} \mid \mathcal{F} \bullet \mathcal{F} \mid \mathcal{F}\backslash\mathcal{F} \tag{5}$$

In this chapter we will explore a broad landscape of categorial type logics. On the semantic level, we are interested in a uniform model theory that naturally accommodates the subtle variations in categorial inference we want to study. A suitable level of abstraction can be obtained by viewing the categorial connectives as *modal* operators, and interpreting the type formulae in the powerset algebra of Kripke-style relational structures. The frame-based semantics for categorial logics is developed in [Došen 92]. As will become clear later on, it extends smoothly to the generalized and mixed architectures that form the core of this chapter. The ‘modal’ semantics also offers a suitable basis for comparison of the categorial systems with the feature-based grammar architectures studied in Chapter Eight.

A modal frame, in general, is a set of ‘worlds’ W together with an $n + 1$ -ary ‘accessibility relation’ R for the n -ary modal operators. In the case of the binary categorial connectives, we interpret with respect to ternary relational structures and consider frames $\langle W, R^3 \rangle$. The domain W is to be thought of here as the set of *linguistic resources* (or: signs, form-meaning complexes of linguistic

information). The ternary accessibility relation R models the core notion of grammatical composition. We obtain a model by adding a valuation v assigning subsets $v(p)$ of W to prime formulae p and satisfying the clauses of Def 2.1 for compound formulae.

Definition 2.1 *Frame semantics: interpretation of compound formulae.*

$$\begin{aligned} v(A \bullet B) &= \{x \mid \exists y \exists z [Rxyz \ \& \ y \in v(A) \ \& \ z \in v(B)]\} \\ v(C/B) &= \{y \mid \forall x \forall z [(Rxyz \ \& \ z \in v(B)) \Rightarrow x \in v(C)]\} \\ v(A \setminus C) &= \{z \mid \forall x \forall y [(Rxyz \ \& \ y \in v(A)) \Rightarrow x \in v(C)]\} \end{aligned}$$

Notice that the categorial vocabulary is highly restricted in its expressivity. In contrast with standard Modal Logic, where the modal operators interact with the usual Boolean connectives, the formula language of the type logics we are considering here is *purely* modal. In §?? we will consider the addition of Boolean operations to the basic categorial language of (5).

We are interested in characterizing a relation of derivability between formulae such that $A \rightarrow B$ is provable iff $v(A) \subseteq v(B)$ for all valuations v over ternary frames. Consider the deductive system **NL**, given by the basic properties of the derivability relation REFL and TRANS, together with the so-called residuation laws RES establishing the relation between \bullet and the two implications $/, \setminus$ with respect to derivability. Prop 2.3 states the essential soundness and completeness result with respect to the frame semantics. (We write ' $\mathcal{L} \vdash A \rightarrow B$ ' for ' $A \rightarrow B$ is provable in logic \mathcal{L} ').

Definition 2.2 *The pure logic of residuation NL ([Lambek 61]).*

$$\begin{aligned} (\text{REFL}) \quad A \rightarrow A \quad (\text{TRANS}) \quad \text{from } A \rightarrow B \text{ and } B \rightarrow C, \text{ infer } A \rightarrow C \\ (\text{RES}) \quad A \rightarrow C/B \quad \text{iff} \quad A \bullet B \rightarrow C \quad \text{iff} \quad B \rightarrow A \setminus C \end{aligned}$$

Proposition 2.3 ([Došen 92]). $\text{NL} \vdash A \rightarrow B$ iff $v(A) \subseteq v(B)$ for every valuation v on every ternary frame.

The proof of the (\Rightarrow) soundness part is by induction on the length of the derivation of $A \rightarrow B$. For the (\Leftarrow) completeness direction, one uses a simple *canonical* model, which effectively falsifies non-theorems. To show that the canonical model is adequate, one proves a Truth Lemma to the effect that, for any formula ϕ , $\mathcal{M}_K, A \models \phi$ iff $\text{NL} \vdash A \rightarrow \phi$. Due to the Truth Lemma we have that if $\text{NL} \not\vdash A \rightarrow B$, then $A \in v_K(A)$ but $A \notin v_K(B)$, so $v_K(A) \not\subseteq v_K(B)$.

Definition 2.4 Define the canonical model as $\mathcal{M}_K = \langle W_K, R_K^3, v_K \rangle$, where

- (i) W_K is the set of formulae \mathcal{F}
- (ii) $R_K^3(A, B, C)$ iff $\text{NL} \vdash A \rightarrow B \bullet C$
- (iii) $A \in v_K(p)$ iff $\text{NL} \vdash A \rightarrow p$

STRUCTURAL POSTULATES, CONSTRAINTS ON FRAMES. In §1, we gave a deconstruction of the notion of grammatical composition into a fixed 'logical' component and a variable 'structural' component. The pure logic of residuation **NL** captures the fixed logical component: the completeness result of Prop 2.3 puts no interpretive constraints whatsoever on the grammatical composition relation. Let us turn now to the resource management component.

Starting from **NL** one can unfold a landscape of categorial type logics by gradually relaxing structure sensitivity in a number of linguistically relevant dimensions. Consider the dimensions of linear precedence (order sensitivity) and immediate dominance (constituent sensitivity). Adding structural postulates licensing associative or commutative resource management (or both) to the pure logic of residuation, one obtains the systems **L**, **NLP**, and **LP**. In order to maintain completeness in the presence of these structural postulates, one has to impose restrictions on the interpretation of the grammatical composition relation R^3 . Below we give the postulates of Associativity

and Commutativity with the corresponding frame constraints. The completeness result of Prop 2.3 is then extended to the stronger logics by restricting the attention to the relevant classes of frames.

Definition 2.5 *Structural postulates and their frame conditions* ($\forall x, y, z, u \in W$).

$$\begin{array}{ll} \text{(ASS)} & (A \bullet B) \bullet C \longleftrightarrow A \bullet (B \bullet C) \quad \exists t. Rtxy \ \& \ Rutz \Leftrightarrow \exists v. Rvyz \ \& \ Ru xv \\ \text{(COMM)} & A \bullet B \rightarrow B \bullet A \quad Rxyz \Leftrightarrow Rxzy \end{array}$$

Proposition 2.6 ([Došen 92]). $\mathbf{L}, \mathbf{NLP}, \mathbf{LP} \vdash A \rightarrow B$ iff $v(A) \subseteq v(B)$ for every valuation v on every ternary frame satisfying (ASS), (COMM), (ASS)+(COMM), respectively.

CORRESPONDENCE THEORY. In the remainder of this chapter, we will consider more dimensions of linguistic structuring than those affected by the Associativity and Commutativity postulates. In [Kurtonina 95] it is shown that one can use the tools of modal Correspondence Theory ([van Benthem 84]) to generalize the completeness results discussed above to these other dimensions. A useful class of structural postulates with pleasant completeness properties is characterized in Def 2.7. The frame conditions for structural postulates of the required weak Sahlqvist form can be effectively computed using the Sahlqvist-Van Benthem algorithm as discussed in [Kurtonina 95].

Definition 2.7 *Weak Sahlqvist Axioms.* A weak Sahlqvist axiom is an arrow of the form $\phi \rightarrow \psi$ where ϕ is a pure product formula, associated in any order, without repetition of proposition letters, and ψ is a pure product formula containing at least one \bullet , all of whose atoms occur in ϕ .

Proposition 2.8 *Sahlqvist Completeness* ([Kurtonina 95]). If P is a weak Sahlqvist axiom, then (i) $\mathbf{NL}+P$ is frame complete for the first order frame condition corresponding to P , and (ii) $\mathcal{L}+P$ has a canonical model whenever \mathcal{L} does.

SPECIALIZED SEMANTICS. As remarked above, the choice for the modal frame semantics is motivated by the desire to have a *uniform* interpretation for the extended and mixed categorial architectures that form the core of this chapter. Grammatical composition is modeled in an abstract way, as a relation between grammatical processes. There is no trace, in this view, of what one could call ‘syntactic representationalism’. As a matter of fact, the relational view on composition does not even require that for resources $y, z \in W$ there will always be a resource x such that $Rxyz$ (Existence), or if such an x exists, that it be unique (Uniqueness).

For many *individual* systems in the categorial hierarchy, completeness results have been obtained for more concrete models. The ‘dynamic’ interpretation of Lambek calculus interprets formulae with respect to *pairs* of points — transitions between information states. The \bullet connective, in this setting, is seen as relational composition. In [Andréka & Mikulás 94], \mathbf{L} is shown to be complete for this interpretation. In groupoid semantics, one considers structures $\langle W, \cdot \rangle$, which can be seen as specializations of the composition relation R^3 : one now reads $Rxyz$ as $x = y \cdot z$, where ‘ \cdot ’ is an arbitrary binary *operation*. Formulae are interpreted in the powerset algebra over these structures, with the simplified interpretation clauses of (6) for the connectives, because the properties of ‘ \cdot ’ now guarantee Existence and Uniqueness.

$$\begin{array}{ll} v(A \bullet B) & = \{x \cdot y \mid x \in v(A) \ \& \ y \in v(B)\} \\ v(C/B) & = \{x \mid \forall y \in v(B) \ x \cdot y \in v(C)\} \\ v(A \setminus C) & = \{y \mid \forall x \in v(A) \ x \cdot y \in v(C)\} \end{array} \quad (6)$$

In the groupoid setting, options for resource management can be realized by attributing associativity and/or commutativity properties to the groupoid operation. Notice that the groupoid models are inappropriate if one wants to consider ‘one-directional’ structural postulates (e.g. one half

of the Associativity postulate, $A \bullet (B \bullet C) \rightarrow (A \bullet B) \bullet C$, allowing restructuring of left-branching structures), unless one is willing to reintroduce abstractness in the form of a partial order on the resources W . See [Došen 92, Buszkowski 86] and Chapter Twelve for discussion.

Even more concrete are the *language models* or free semigroup semantics for \mathbf{L} . In the language models, one takes W as V^+ (non-empty strings over the vocabulary) and \cdot as string concatenation. This type of semantics turns out to be too specialized for our purposes: whereas [Pentus 94], with a quite intricate proof, has been able to establish completeness of \mathbf{L} with respect to the free semigroup models, there is an *incompleteness* result for \mathbf{NL} with respect to the corresponding free non-associative structures, viz. finite tree models. See [Venema 94] for discussion.

GENERAL MODELS VERSUS SPECIFIC GRAMMARS. In the discussion so far we have studied type-logical derivability in completely general terms, abstracting away from language-specific grammar specifications. Let us see then how we can relativize the general notions so as to take actual grammar specification into account. In accordance with the categorical tenet of *radical lexicalism*, we assume that the grammar for a language L is given by the conjunction of the general type logic \mathcal{L} with a language-specific lexicon $\text{LEX}(L)$. The lexicon itself is characterized in terms of a type assignment function $f : V_L \mapsto \mathcal{P}(\mathcal{F})$, stipulating the primitive association of lexical resources V_L with their types. (We assume that for all lexical resources $x \in V_L$, the sets $f(x)$ are finite. In the so-called *rigid* categorical grammars, one further restricts the values of f to be *singletons*.)

For a general model $\mathcal{M} = \langle W, R^3, v \rangle$ to qualify as appropriate for $\text{LEX}(L)$, we assume $V_L \subseteq W$, and we require the valuation v to be compatible with lexical type assignment, in the sense that, for all $x \in V_L$, $A \in f(x)$ implies $x \in v(A)$. Given this, we will say that the grammar assigns type B to a non-empty string of lexical resources $x_1 \dots x_n \in V_L^+$, provided there are lexical type specifications $A_i \in f(x_i)$ such that we can deduce B from $\circ(A_1, \dots, A_n)$ in the general type-logic \mathcal{L} . By $\circ(A_1, \dots, A_n)$ we mean any of the possible products of the formulas A_1, \dots, A_n , in that order.

CATEGORICAL COMBINATORS AND CCG. To round off the discussion of the axiomatic presentation, we present the logics \mathbf{NL} , \mathbf{L} , \mathbf{NLP} , \mathbf{LP} with a proof term annotation, following [Lambek 88]. The proof terms — categorical combinators — are motivated by Lambek’s original category-theoretic interpretation of the type logics. The category-theoretic connection is not further explored here, but the combinator proof terms will be used in later sections as compact notation for complete deductions.

Definition 2.9 *Combinator proof terms ([Lambek 88]).* Deductions of the form $f : A \rightarrow B$, where f is a process for deducing B from A .

$$\begin{array}{c}
1_A : A \rightarrow A \qquad \frac{f : A \rightarrow B \quad g : B \rightarrow C}{g \circ f : A \rightarrow C} \\
\\
\frac{f : A \bullet B \rightarrow C}{\beta_{A,B,C}(f) : A \rightarrow C/B} \qquad \frac{f : A \bullet B \rightarrow C}{\gamma_{A,B,C}(f) : B \rightarrow A \setminus C} \\
\\
\frac{g : A \rightarrow C/B}{\beta_{A,B,C}^{-1}(g) : A \bullet B \rightarrow C} \qquad \frac{g : B \rightarrow A \setminus C}{\gamma_{A,B,C}^{-1}(g) : A \bullet B \rightarrow C} \\
\\
\alpha_{A,B,C} : A \bullet (B \bullet C) \longleftrightarrow (A \bullet B) \bullet C : \alpha_{A,B,C}^{-1} \\
\pi_{A,B} : A \bullet B \rightarrow B \bullet A
\end{array}$$

Example 2.10 *Combinator proof terms for rightward functional application, and for leftward type lifting. (We omit the type subscripts where they are clear from context.) In the derivation of lifting, we write RA for $\beta^{-1}(1_{A/B})$.*

$$\frac{1_{A/B} : A/B \rightarrow A/B}{\beta^{-1}(1_{A/B}) : A/B \bullet B \rightarrow A} \quad \frac{RA : A/B \bullet B \rightarrow A}{\gamma(RA) : B \rightarrow (A/B) \setminus A}$$

This presentation makes obvious a variety of methods for creating fragments (subsystems) and extensions: restrict or extend the formula language; remove or add inference rules; remove or add structural postulates. The Ajdukiewicz/Bar-Hillel system [Ajdukiewicz 35, Bar-Hillel 64] appears in this guise as a subsystem lacking the hypothetical reasoning rules β and γ and the permutation rule π , but implicitly countenancing associativity. A more complex example is the rule-based approach of Combinatory Categorical Grammar (CCG, [Steedman 93]) where a finite collection of unary type transitions and binary type combinations (such as Lifting, Application) are postulated as *primitive* rule schemata. Within the CCG framework, Combinatory Logic ([Curry & Feys 58]) is put forward as the general theory for the class of grammatical operations natural languages draw upon. Combinatory Logic in itself, being equivalent with the full Lambda Calculus in its expressivity, is not very informative as to the fine-structure of grammatical inference. A decomposition of the CCG combinators in their logical and structural parts uncovers the hidden assumptions about grammatical resource management and makes it possible to situate the CCG systems within a more articulate landscape of grammatical inference. Comparing the CCG framework with the type-logical approach studied here, one should realize that CCG systems are, by necessity, only approximations of logics such as **L**, **LP**. These logics have been shown to be *not finitely axiomatizable* (see [Zielonka 89] and Chapter Twelve), which means that no finite set of combinators in combination with Modus Ponens can equal their deductive strength.

2.2 Gentzen calculus, cut elimination and decidability

The axiomatic presentation is the proper vehicle for model-theoretic investigation of the logics we have considered: it closely follows the semantics, thus providing a suitable basis for ‘easy’ completeness results. But proof-theoretically the axiomatic presentation has a serious drawback: it does not offer an appropriate basis for proof search. The problematic rule of inference is TRANS, which is used to *compose* type transitions $A \rightarrow B$ and $B \rightarrow C$ into a transition $A \rightarrow C$. A type transition $A \rightarrow C$, in the presence of TRANS, could be effected with the aid of a formula B of which one finds no trace in the conclusion of the TRANS inference. Since there is an infinity of candidate formulae B , exhaustive traversal of the search space for the auxiliary B formula in a TRANS inference is not an option.

For proof-theoretic investigation of the categorial type logics one introduces a Gentzen presentation which is shown to be equivalent to the axiomatic presentation. The main result for the Gentzen calculus (the *Hauptsatz* of [Gentzen 34]) then states that the counterpart of the TRANS rule, the Cut inference, can be eliminated from the logic without affecting the set of derivable theorems. An immediate corollary of this Cut Elimination Theorem is the *subformula property* which limits proof search to the subformulae of the theorem one wants to derive. In the absence of resource-affecting structural rules, decidability follows. The essential results for **L** have been established in [Lambek 58]. They have been extended to the full landscape of type logics in [Kandulski 88, Došen 89].

In the axiomatic presentation, we considered derivability as a relation between formulae, i.e. we considered arrows $A \rightarrow B$ with $A, B \in \mathcal{F}$. In the Gentzen presentation, the derivability relation is stated to hold between a *term* S (the antecedent) and a type formula (the succedent). A Gentzen term is a structured configuration of formulae — a structured database, in the terminology of [Gabbay 94]. The term language is defined inductively as $S ::= \mathcal{F} \mid (S, S)$. The binary structural connective (\cdot, \cdot) in the term language tells you how structured databases Δ_1 and Δ_2 have been put together into a structured database (Δ_1, Δ_2) . The structural connective mimics the logical connective \bullet in the type language. A sequent is a pair (Γ, A) with $\Gamma \in \mathcal{S}$ and $A \in \mathcal{F}$, written as $\Gamma \Rightarrow A$.

To establish the equivalence between the two presentations, we define the formula translation Δ° of a structured database Δ : $(\Delta_1, \Delta_2)^\circ = \Delta_1^\circ \bullet \Delta_2^\circ$, and $A^\circ = A$, for $A \in \mathcal{F}$.

Proposition 2.11 ([Lambek 58]). *For every arrow $f : A \rightarrow B$ there is a Gentzen proof of $A \Rightarrow B$, and for every proof of a sequent $\Gamma \Rightarrow B$ there is an arrow $f : \Gamma^\circ \rightarrow B$.*

Definition 2.12 NL: *Gentzen presentation. Sequents $S \Rightarrow \mathcal{F}$ where $S ::= \mathcal{F} \mid (S, S)$. We write $\Gamma[\Delta]$ for a term Γ containing a distinguished occurrence of the subterm Δ . (The distinguished occurrences in premise and conclusion of an inference rule are supposed to occupy the same position within Γ .)*

$$\begin{array}{c}
[\text{Ax}] \frac{}{A \Rightarrow A} \quad \frac{\Delta \Rightarrow A \quad \Gamma[A] \Rightarrow C}{\Gamma[\Delta] \Rightarrow C} [\text{Cut}] \\
[\text{/R}] \frac{(\Gamma, B) \Rightarrow A}{\Gamma \Rightarrow A/B} \quad \frac{\Delta \Rightarrow B \quad \Gamma[A] \Rightarrow C}{\Gamma[(A/B, \Delta)] \Rightarrow C} [\text{/L}] \\
[\text{\R}] \frac{(B, \Gamma) \Rightarrow A}{\Gamma \Rightarrow B \setminus A} \quad \frac{\Delta \Rightarrow B \quad \Gamma[A] \Rightarrow C}{\Gamma[(\Delta, B \setminus A)] \Rightarrow C} [\text{\L}] \\
[\bullet\text{L}] \frac{\Gamma[(A, B)] \Rightarrow C}{\Gamma[A \bullet B] \Rightarrow C} \quad \frac{\Gamma \Rightarrow A \quad \Delta \Rightarrow B}{(\Gamma, \Delta) \Rightarrow A \bullet B} [\bullet\text{R}]
\end{array}$$

As was the case for the axiomatic presentation of Def 2.2, the Gentzen architecture of Def 2.12 consists of three components: (i) [Ax] and [Cut] capture the basic properties of the derivability relation ‘ \Rightarrow ’: reflexivity and contextualized transitivity for the Cut rule, (ii) each connective comes with two *logical rules*: a rule of use introducing the connective to the left of ‘ \Rightarrow ’ and a rule of proof introducing it on the right of ‘ \Rightarrow ’, finally (iii) there is a block of *structural rules*, empty in the case of **NL**, with different packages of structural rules resulting in systems with different resource management properties. (We should note here that sometimes the Cut rule is counted among the structural rules. We will reserve the term ‘structural rule’ for the Gentzen counterpart of the structural postulates governing the resource management properties of the composition operation.)

STRUCTURAL RULES. Structural postulates, in the axiomatic presentation, have been presented as transitions $A \rightarrow B$ where A and B are constructed out of formula variables p_1, \dots, p_n and the logical connective \bullet . For corresponding structure variables $\Delta_1, \dots, \Delta_n$ and the structural connective (\cdot, \cdot) , define the structural equivalent $\sigma(A)$ of a formula A : $\sigma(p_i) = \Delta_i$, $\sigma(A \bullet B) = (\sigma(A), \sigma(B))$. The transformation of structural postulates into Gentzen rules allowing Cut Elimination is then straightforward: a postulate $A \rightarrow B$ translates as the Gentzen rule (7):

$$\frac{\Gamma[\sigma(B)] \Rightarrow C}{\Gamma[\sigma(A)] \Rightarrow C} \quad (7)$$

To obtain the logics **L**, **NLP**, **LP** from **NL**, one thus adds the structural rules of Associativity and/or Permutation. Such additions result in less fine-grained notions of linguistic inference, where structural discrimination with respect to the dimensions of dominance and/or precedence is lost, as discussed above. (The double line in [A] stands for a two-way inference.)

$$\frac{\Gamma[(\Delta_2, \Delta_1)] \Rightarrow A}{\Gamma[(\Delta_1, \Delta_2)] \Rightarrow A} [\text{P}] \quad \frac{\Gamma[((\Delta_1, \Delta_2), \Delta_3)] \Rightarrow A}{\Gamma[(\Delta_1, (\Delta_2, \Delta_3))] \Rightarrow A} [\text{A}] \quad (8)$$

SUGARING. For the logics **L** and **LP** where \bullet is associative, resp. associative and commutative, explicit application of the structural rules is generally compiled away by means of syntactic sugaring of the sequent language. Antecedent terms then take the form of sequences of formulae

A_1, \dots, A_n where the comma is now of variable arity, rather than a binary connective. Reading these antecedents as sequences, one avoids explicit reference to the Associativity rule; reading them as multisets, one also makes Permutation implicit.

Definition 2.13 *Sugared Gentzen presentation: implicit structural rules.* Sequents $S \Rightarrow \mathcal{F}$ where $S ::= \mathcal{F} \mid \mathcal{F}, S$. **L**: implicit Associativity, interpreting S as a sequence. **LP**: implicit Associativity+Permutation, interpreting S as a multiset. (The context variables Γ, Γ' can be empty.)

$$\begin{array}{c}
[\text{Ax}] \frac{}{A \Rightarrow A} \quad \frac{\Delta \Rightarrow A \quad \Gamma, A, \Gamma' \Rightarrow C}{\Gamma, \Delta, \Gamma' \Rightarrow C} [\text{Cut}] \\
[\text{/R}] \frac{\Delta, B \Rightarrow A}{\Delta \Rightarrow A/B} \quad \frac{\Delta \Rightarrow B \quad \Gamma, A, \Gamma' \Rightarrow C}{\Gamma, A/B, \Delta, \Gamma' \Rightarrow C} [\text{/L}] \\
[\text{\R}] \frac{B, \Delta \Rightarrow A}{\Delta \Rightarrow B \setminus A} \quad \frac{\Delta \Rightarrow B \quad \Gamma, A, \Gamma' \Rightarrow C}{\Gamma, \Delta, B \setminus A, \Gamma' \Rightarrow C} [\text{\L}] \\
[\bullet\text{L}] \frac{\Gamma, A, B, \Gamma' \Rightarrow C}{\Gamma, A \bullet B, \Gamma' \Rightarrow C} \quad \frac{\Delta \Rightarrow A \quad \Delta' \Rightarrow B}{\Delta, \Delta' \Rightarrow A \bullet B} [\bullet\text{R}]
\end{array}$$

CUT ELIMINATION AND DECIDABILITY. A categorial version of Gentzen's *Hauptsatz* is the core of [Lambek 58], who proves Cut Elimination for **L**, on the basis of the 'sugared' presentation introduced in Def 2.13. In [Došen 89] the result is extended to the full landscape of categorial logics, using the structured term representation of antecedent databases, and explicit structural rules. It is important to carefully distinguish between an *admissible* rule of inference versus a *derived* one. We will see examples of derived rules of inference in Prop 2.18: as the name indicates, one can deduce the derived inference rules using the basic logical rules for the connectives. The Cut rule cannot be so derived — it does not mention any logical connectives. But is admissible in the sense that it does not increase the set of theorems that can already be derived using just the logical rules of inference.

Proposition 2.14 *Cut Elimination ([Lambek 58, Došen 89]).* The Cut rule is admissible in **NL**, **L**, **NLP**, **LP**: every theorem has a cut-free proof.

Below we present the general strategy for the cut elimination transformation, so that the reader can check how the various extensions of the type-logical vocabulary we will consider in the remainder of this chapter can be accommodated under the general cases of the elimination schema.

CUT ELIMINATION ALGORITHM. The proof of the admissibility of the Cut rule is a constructive algorithm for a stepwise transformation of a derivation involving Cut inferences into a Cut-free derivation. Eliminability of the Cut rule is proved by induction on the complexity d of Cut inferences, measured in the number of connective occurrences. For the Cut rule of Def 2.13, we have the following schema, with Cut complexity d defined as $d(\text{Cut}) = d(\Delta) + d(\Gamma) + d(\Gamma') + d(A) + d(B)$.

$$\frac{\Delta \Rightarrow A \quad \Gamma, A, \Gamma' \Rightarrow B}{\Gamma, \Delta, \Gamma' \Rightarrow B} \text{Cut} \tag{9}$$

The targets for the elimination algorithm are instances of Cut which have themselves been derived without using the Cut rule. It is shown that in the derivation in question such a Cut inference can be replaced by one or two Cuts of lower degree. One repeats the process until all Cuts have been removed. The following main cases can be distinguished.

Case 1 The base case of the recursion: one of the Cut premises is an Axiom. In this case the other premise is identical to the conclusion, and the application of Cut can be pruned.

Case 2 Permutation conversions. In these cases, the active formula in the left or right premise of Cut is *not* the Cut formula. One shows that the logical rule introducing the main connective of the active formula and the Cut rule can be permuted, pushing the Cut inference upwards, with a decrease in degree because a connective is now introduced lower in the proof. (Explicit structural rules for the structured antecedent representation assimilate to this case: the Cut rule is permuted upwards over the structural rule.)

Case 3 Principal Cuts. The active formula in the left and right premise of Cut make up the Cut formula A . Here one reduces the degree by splitting the Cut formula up into its two immediate subformulae, and applying Cuts on these.

Example 2.15 *Case 2. The active formula in the left Cut premise is A'/A'' . The Cut rule is moved upwards, permuting with the $[L]$ logical inference.*

$$\frac{\frac{\frac{\Delta'' \Rightarrow A'' \quad \Delta, A', \Delta' \Rightarrow A}{\Delta, A'/A'', \Delta'', \Delta' \Rightarrow A} /L \quad \Gamma, A, \Gamma' \Rightarrow B}{\Gamma, \Delta, A'/A'', \Delta'', \Delta', \Gamma' \Rightarrow B} \text{Cut} \quad \sim \quad \frac{\frac{\Delta, A', \Delta' \Rightarrow A \quad \Gamma, A, \Gamma' \Rightarrow B}{\Gamma, \Delta, A', \Delta', \Gamma' \Rightarrow B} \text{Cut} \quad \frac{\Delta'' \Rightarrow A''}{\Gamma, \Delta, A'/A'', \Delta'', \Delta', \Gamma' \Rightarrow B} /L}{\Gamma, \Delta, A'/A'', \Delta'', \Delta', \Gamma' \Rightarrow B} \text{Cut}}$$

Example 2.16 *Case 3. Principal Cut on A'/A'' . The Cut inference is replaced by two Cuts, on the subformulae A' and A'' .*

$$\frac{\frac{\frac{\Delta, A'' \Rightarrow A' \quad \Delta' \Rightarrow A'' \quad \Gamma, A', \Gamma' \Rightarrow B}{\Gamma, A'/A'', \Delta', \Gamma' \Rightarrow B} /L \quad \frac{\Delta \Rightarrow A'/A''}{\Gamma, \Delta, \Delta', \Gamma' \Rightarrow B} /R}{\Gamma, \Delta, \Delta', \Gamma' \Rightarrow B} \text{Cut} \quad \sim \quad \frac{\frac{\Delta, A'' \Rightarrow A' \quad \Gamma, A', \Gamma' \Rightarrow B}{\Gamma, \Delta, A'', \Gamma' \Rightarrow B} \text{Cut} \quad \frac{\Delta' \Rightarrow A''}{\Gamma, \Delta, \Delta', \Gamma' \Rightarrow B} \text{Cut}}{\Gamma, \Delta, \Delta', \Gamma' \Rightarrow B} \text{Cut}}$$

DECIDABILITY, SUBFORMULA PROPERTY. In the case of **NL**, **L**, **NLP**, **LP**, the Cut Elimination theorem immediately gives a decision procedure for theoremhood. One searches for a cut-free proof in a backward chaining manner, working from conclusion to premises. Every logical rule of inference removes a connective, breaking the selected active formula up into its immediate subformulae. The number of connectives of the goal sequent is finite. Exhaustive traversal of the finite cut-free search space will either produce a proof (a derivation tree the leaves of which are all instances of the Axiom schema), or it will fail to do so.

The important point about Cut Elimination and decidability is not so much to avoid the Cut rule altogether, but to restrict the attention to ‘safe’ cuts — instances of the Cut rule that do not affect the finiteness of the search space. The astute reader will have noticed that the left rules for the implications $A \setminus B$ (B/A) are in fact *compiled* Cut inferences on the basis of the subtypes A , B and Modus Ponens. These compiled Cuts are innocent: they preserve the complexity-decreasing property of the inference rules which guarantees decidability. The compilation of $[\setminus L]$ can be found below.

$$\frac{\frac{\frac{\Delta \Rightarrow B \quad \frac{B \Rightarrow B \quad A \Rightarrow A}{B, B \setminus A \Rightarrow A} \setminus L}{\Delta, B \setminus A \Rightarrow A} [\text{Cut}] \quad \Gamma, A, \Gamma' \Rightarrow C}{\Gamma, \Delta, B \setminus A, \Gamma' \Rightarrow C} [\text{Cut}] \quad \sim \quad \frac{\Delta \Rightarrow B \quad \Gamma, A, \Gamma' \Rightarrow C}{\Gamma, \Delta, B \setminus A, \Gamma' \Rightarrow C} \setminus L}{\Gamma, \Delta, B \setminus A, \Gamma' \Rightarrow C} \setminus L} \quad (10)$$

NATURAL DEDUCTION. As a final item on the list of presentation formats for categorial derivations, Def 2.17 gives the official definition of the Natural Deduction format used in §1. This style of presentation has Elimination rules and Introduction rules for the logical connectives: the Cut rule is not a part of the Natural Deduction format.

For the equivalence of the sequent and natural deduction styles the reader can turn to [Girard e.a. 89], where one finds explicit mappings relating the two presentations. The mapping from Gentzen proofs to natural deductions is *many-to-one* — there may be a number of Gentzen derivations for one and the same natural deduction. In this sense, natural deduction captures the ‘essence’ of a proof better than a Gentzen derivation, which allows irrelevant permutation alternatives in deriving a theorem. We will be in a better position to assess this spurious type of non-determinism of the sequent calculus after discussing the Curry-Howard interpretation of categorial deductions, which gives a precise answer to the question as to which derivations are ‘essentially the same’. See Def 3.5 and §?? for discussion.

Definition 2.17 *Natural deduction. Sequent-style presentation. Notation:* $\Gamma \vdash A$ *for a deduction of the formula* A *from a configuration of undischarged assumptions* Γ . *Elimination/Introduction rules for* **NL**. *Structural rules as in* (8).

$$\begin{array}{c}
A \vdash A \\
\\
\begin{array}{cc}
\text{[I]} \frac{(\Gamma, B) \vdash A}{\Gamma \vdash A/B} & \frac{\Gamma \vdash A/B \quad \Delta \vdash B}{(\Gamma, \Delta) \vdash A} \text{[E]} \\
\text{[I]} \frac{(B, \Gamma) \vdash A}{\Gamma \vdash B \setminus A} & \frac{\Gamma \vdash B \quad \Delta \vdash B \setminus A}{(\Gamma, \Delta) \vdash A} \text{[E]} \\
\text{[•I]} \frac{\Gamma \vdash A \quad \Delta \vdash B}{(\Gamma, \Delta) \vdash A \bullet B} & \frac{\Delta \vdash A \bullet B \quad \Gamma[(A, B)] \vdash C}{\Gamma[\Delta] \vdash C} \text{[•E]}
\end{array}
\end{array}$$

2.3 Discussion: options for resource management

In the previous sections, we have introduced the technical apparatus that is needed for a proper appreciation of the logics **NL**, **L**, **NLP**, **LP**. Let us turn now to the linguistic motivation for the different resource management regimes these logics represent. In order to compare the strengths and weaknesses of these individual systems, Prop 2.18 gives a useful inventory of characteristic theorems and derived rules of inference for the logics in question. We leave their proof to the reader, who can test his or her understanding of the axiomatic and Gentzen presentations in deriving them.

Proposition 2.18 *Characteristic theorems and derived inference rules for* **NL** *(1–6);* **L** *(7–11), plus* (1–6); **NLP** *(12–14), plus* (1–6); **LP** *(15), plus* (1–14).

- | | |
|---|--|
| 1. <i>Application:</i> | $A/B \bullet B \rightarrow A, B \bullet B \setminus A \rightarrow A$ |
| 2. <i>Co-application:</i> | $A \rightarrow (A \bullet B)/B, A \rightarrow B \setminus (B \bullet A)$ |
| 3. <i>Monotonicity •:</i> | <i>if</i> $A \rightarrow B$ <i>and</i> $C \rightarrow D$, <i>then</i> $A \bullet C \rightarrow B \bullet D$ |
| 4. <i>Isotonicity</i> $/C, C \setminus \cdot$: | <i>if</i> $A \rightarrow B$, <i>then</i> $A/C \rightarrow B/C$ <i>and</i> $C \setminus A \rightarrow C \setminus B$ |
| 5. <i>Antitonicity</i> $C/\cdot, \cdot \setminus C$: | <i>if</i> $A \rightarrow B$, <i>then</i> $C/B \rightarrow C/A$ <i>and</i> $B \setminus C \rightarrow A \setminus C$ |
| 6. <i>Lifting:</i> | $A \rightarrow B/(A \setminus B), A \rightarrow (B/A) \setminus B$ |
| 7. <i>Geach (main functor):</i> | $A/B \rightarrow (A/C)/(B/C), B \setminus A \rightarrow (C \setminus B) \setminus (C \setminus A)$ |
| 8. <i>Geach (secondary functor):</i> | $B/C \rightarrow (A/B) \setminus (A/C), C \setminus B \rightarrow (C \setminus A)/(B \setminus A)$ |
| 9. <i>Composition:</i> | $A/B \bullet B/C \rightarrow A/C, C \setminus B \bullet B \setminus A \rightarrow C \setminus A$ |
| 10. <i>Restructuring:</i> | $(A \setminus B)/C \longleftrightarrow A \setminus (B/C)$ |
| 11. <i>(De)Currying:</i> | $A/(B \bullet C) \longleftrightarrow (A/C)/B, (A \bullet B) \setminus C \longleftrightarrow B \setminus (A \setminus C)$ |
| 12. <i>Permutation:</i> | <i>if</i> $A \rightarrow B \setminus C$ <i>then</i> $B \rightarrow A \setminus C$ |
| 13. <i>Exchange:</i> | $A/B \longleftrightarrow B \setminus A$ |
| 14. <i>Preposing/Postposing:</i> | $A \rightarrow B/(B/A), A \rightarrow (A \setminus B) \setminus B$ |
| 15. <i>Mixed Composition:</i> | $A/B \bullet C \setminus B \rightarrow C \setminus A, B/C \bullet B \setminus A \rightarrow A/C$ |

Items (1) to (6) are valid in the most discriminating logic **NL**. As shown in [Došen 89], the combination of (1)–(5) provides an alternative way of characterizing $(\bullet, /)$ and (\bullet, \backslash) as residuated pairs, i.e. one can replace the RES inferences of Def 2.2 by (1)–(5). The reader with a background in category theory recognizes the adjointness (1–2) and functoriality (3–5) laws. Lifting is the closest one can get to (2) in ‘product-free’ type languages, i.e. type languages where the role of the product operator (generally left implicit) is restricted to glue together purely implicational types on the left-hand side of the arrow. Items (7) to (11) mark the transition to **L**: their derivation involves the structural postulate of associativity for \bullet . Rule (12) is characteristic for systems with a commutative \bullet , **NLP** and **LP**. From (12) one immediately derives the collapse of the implications $/$ and \backslash , (13). As a result of this collapse, one gets variants of the earlier theorems obtained by substituting subtypes of the form A/B by $B\backslash A$ or vice versa. Examples are (14), an **NLP** variant of Lifting, or (15), an **LP** variant of Composition.

THE PURE LOGIC OF RESIDUATION. Let us look first at the most discriminating logic in the landscape, **NL**. In the absence of structural postulates for \bullet , grammatical inference is fully sensitive to both the horizontal and the vertical dimensions of linguistic structure: linear ordering and hierarchical grouping. As in classical Ajdukiewicz style categorial grammar, Application is the basic reduction law for this system. But the capacity for *hypothetical reasoning* already greatly increases the inferential strength of **NL** in comparison with the pure application fragment. The principles of Argument Lowering (e.g. $(s/(np\backslash s))\backslash s \rightarrow np\backslash s$) and Value Raising (e.g. $np/n \rightarrow (s/(np\backslash s))/n$), introduced as primitive postulates in [Partee & Rooth 83], turn out to be generally valid type change schemata, derivable from the combination of Lifting and the Isotonicity/Antitonicity laws for the implications. These type-changing laws play an important role in the semantic investigation of categorial type systems, as we will see in §3. On a more general level, it is pointed out in [Lambek 88] that Lifting is a closure operation, as it obeys the defining principles (11). (We write A^B for either $B/(A\backslash B)$ or $(B/A)\backslash B$.)

$$A \rightarrow A^B \quad (A^B)^B \rightarrow A^B \quad \text{from } A \rightarrow C, \text{ infer } A^B \rightarrow C^B \quad (11)$$

ASSOCIATIVITY AND FLEXIBLE CONSTITUENCY. An essential limitation of the pure residuation logic is its rigid concept of constituency — a property which **NL** shares with conventional phrase structure grammars. The revival of interest in categorial grammar was inspired in the first place by a more flexible notion of constituent structure, depending on **L** theorems such as the Geach laws, Functional Composition, or its recursive generalization. These Geach and Composition principles are formulated as *implicational* laws, but with the interpretation of the type-logical connectives we have been assuming, the implicational laws and the product versions of the structural postulates are interderivable.

Example 2.19 *Deriving (one half of) Associativity from (one directional instantiation of) Geach. (We write \mathfrak{b} for the left-division variant of principle (7) in Prop 2.18, and use X as an abbreviation for $A \bullet ((A\backslash(A \bullet B)) \bullet ((A \bullet B)\backslash((A \bullet B) \bullet C)))$.)*

$$\frac{\frac{\frac{\mathfrak{b} : A\backslash B \rightarrow (C\backslash A)\backslash(C\backslash B)}{\gamma^{-1}(\mathfrak{b}) : C\backslash A \bullet A\backslash B \rightarrow C\backslash B}}{(\dagger) \gamma^{-1}(\gamma^{-1}(\mathfrak{b})) : C \bullet (C\backslash A \bullet A\backslash B) \rightarrow B}}{\frac{\frac{A \rightarrow A \quad \frac{(2) : B \rightarrow A\backslash(A \bullet B) \quad (2) : C \rightarrow (A \bullet B)\backslash((A \bullet B) \bullet C)}{(3) : B \bullet C \rightarrow (A\backslash(A \bullet B)) \bullet ((A \bullet B)\backslash((A \bullet B) \bullet C))}}{(3) : A \bullet (B \bullet C) \rightarrow X} \quad (\dagger) : X \rightarrow (A \bullet B) \bullet C}{A \bullet (B \bullet C) \rightarrow (A \bullet B) \bullet C}}$$

Associative resource management makes the grammar logic insensitive to hierarchical constituent structure: derivability of a sequent $\Gamma \Rightarrow A$ is preserved under arbitrary rebracketings of the antecedent assumptions Γ , a property which is referred to as the *structural completeness* of **L** ([Buszkowski 88]).

derivation of (12), as pointed out in [Steedman 93], where this type of example is traced back to [Lambek 61].

*(the mathematician) whom Gottlob admired Kazimierz and Jim detested (12)

DISCONTINUOUS DEPENDENCIES AND RESTRICTED COMMUTATIVITY. The discussion above shows that **L** is too strong in that it fully ignores constituent information. But at the same time, the order sensitivity of this logic makes it too weak to handle discontinuous dependencies. A case in point are the crossed dependencies of the Dutch verb cluster. In the example below, the verb raising trigger ‘wil’ has to combine with the infinitival ‘voeren’ before the latter (a transitive verb) combines with its direct object.

Example 2.22 *Dutch verb clusters via Mixed Composition. (We write iv for infinitival verb phrases, vp for tensed ones.)*

(dat Marie) de nijlpaarden (np) wil (vp/iv) voeren ($np\ iv$)
 ‘(that Mary) the hippos wants feed’ (= that M. wants to feed the hippos)
 $vp/iv, np\ iv \Rightarrow np\ vp$ (Mixed Composition)
 $A/B \rightarrow (C\ A)/(C\ B)$ (Mixed Geach, schematically)

In order to form the cluster ‘wil voeren’ in such a way that it ‘inherits’ the arguments of the embedded infinitive, composition laws (or their Geach generalizations) have been proposed ([Steedman 84, Moortgat 88]) that would combine functors with conflicting directionality requirements, so-called Mixed Composition. Clearly, with these laws, one goes beyond the inferential capacity of **L**. As the reader can check with the aid of Ex 2.19, the product counterpart of the mixed Geach transition of Ex 2.22 is $A \bullet (B \bullet C) \rightarrow B \bullet (A \bullet C)$, which together with Associativity introduces a contextualized form of Commutativity. The permutation side effects of Mixed Composition cause a damaging loss of control over the grammatical resources. Not surprisingly, then, the introduction of such combination schemata went hand in hand with the formulation of *extralogical* control principles. A more attractive alternative will be presented in Ex 4.4.

CONCLUSION. Let us summarize this discussion. The individual simple Lambek systems each have their merits and their limitations when it comes to grammatical analysis. As a grammar writer, one would like to exploit the inferential capacities of a *combination* of different systems. Importing theorems from a system with more relaxed resource management into a logic with a higher degree of structural discrimination is not a viable strategy: it globally affects sensitivity for the relevant structural parameter of the more discriminating logic. In §4.1 we will develop a logical framework supporting a truly ‘mixed’ style of categorial inference. Structural collapse is avoided by moving to a multimodal architecture which is better adapted to deal with the fine-structure of grammatical composition. But first we discuss an aspect of grammatical inference which is of crucial importance for the categorial architecture but which has been ignored so far: the syntax-semantics interface.

3 The syntax-semantics interface: proofs and readings

Categorial type logics offer a highly transparent view on the relation between form and meaning: semantic interpretation can be read off directly from the proof which establishes the well-formedness (derivability) of an expression. The principle of compositionality (see Chapters One and Seven) is realized in a particularly stringent, purely deductive form, leaving no room for rule-to-rule stipulated meaning assignment.

In §1 we noticed that the categorial program ultimately has its ancestry in Russell’s theory of types. In the original ‘Polish’ version of the program, categorial types were viewed simultaneously in the

syntactic and in the semantic dimension. This unified perspective was lost in subsequent work: Lambek developed categorial calculi as theories of *syntactic* types, and Curry advocated the application of his *semantic* types of functionality in natural language analysis — a development which led up to Montague’s use of type theory. The divergence can be traced back to [Jakobson 61], where [Curry 61] in fact criticizes [Lambek 61] for introducing the structural dimension of grammatical composition in his category concept. These divergent lines of research were brought together again in [van Benthem 83], who established the connection between Lambek’s categorial framework and the Curry-Howard ‘formulas-as-types’ program.

In the logical setting, the Curry-Howard program takes the form of an *isomorphism* between (Natural Deduction) proofs in the Positive Intuitionistic Propositional Logic and terms of the λ calculus. In the categorial application, one is interested in the Curry-Howard mapping as a *correspondence* rather than an isomorphism, in the sense that derivations for the various categorial logics are all associated with **LP** term recipes. The system **LP**, in this sense, plays the role of a general semantic composition language which abstracts away from syntactic fine-structure. As we have seen in §2, the form dimension of grammatical composition can be profitably studied in the context of the *frame semantics* for the type formulae: on that level, the structural postulates regulating sub-**LP** resource management naturally find their interpretation in terms of frame constraints.

The emphasis in this section is on the *limited semantic expressivity* of the categorial languages. With respect to the original intuitionistic terms, the **LP** fragment obeys linearity constraints reflecting the resource sensitivity of the categorial logic; moving on to more discriminating systems, the set of derivable readings further decreases. The price one pays for obtaining more fine-grained syntactic discrimination may be the loss of readings one would like to retain from a purely semantic point of view. This tension has played an important role in the development of the field. To regain lost readings one can enrich the logical vocabulary, and introduce more delicate type constructors compatible with both the structural and the semantic aspects of grammatical composition. And one can exploit the division of labour between *lexical* and *derivational* semantics. We discuss this theme in §3.2. In §3.1 we first introduce the necessary technical material, basing the exposition on [van Benthem 91,95, Wansing 92b, Hendriks 93]. Our treatment of term assignment focuses on the Gentzen presentation of the categorial calculi. For a parallel treatment in terms of Natural Deduction, the reader can turn to Chapter Twelve.

3.1 Term assignment for categorial deductions

We start our discussion of semantic term assignment with the system at the top of the categorial hierarchy — the system **LP**. Instead of sequents $A_1, \dots, A_n \Rightarrow B$ we now consider annotated sequents $x_1 : A_1, \dots, x_n : A_n \Rightarrow t : B$ where the type formulae are decorated with terms — distinct x_i for the assumptions and a term t constructed out of these x_i , in ways to be made precise below, for the goal. On the intuitive level, a derivation for an annotated sequent will represent the computation of a denotation recipe t of type B with input parameters x_i of type A_i . Let us specify the syntax and semantics of the language of type formulae and term labels, and define the systematic association of the term labeling with the unfolding of a sequent derivation.

In the case of **LP**, we are considering a type language with formulae $\mathcal{F} ::= \mathcal{A} \mid \mathcal{F} \rightarrow \mathcal{F} \mid \mathcal{F} \circ \mathcal{F}$ (the two implications collapse in the presence of Permutation). The choice of primitive types \mathcal{A} will depend on the application. A common choice would be e (the type of individual objects) and t (the type of truth values). In §3.2, we will encounter more elaborate inventories for the ‘dynamic’ approach to natural language semantics. For semantic interpretation of the type language, we consider frames $F = \{D_A\}_{A \in \mathcal{F}}$ based on some non-empty set E , the domain of discourse. Such frames consist of a family of semantic domains, one for each type $A \in \mathcal{F}$, such that

$$D_{A \circ B} = D_A \times D_B \quad (\text{Cartesian product}) \quad D_{A \rightarrow B} = D_B^{D_A} \quad (\text{Function space}) \quad (13)$$

For the primitive types we can fix $D_e = E$ and $D_t = \{0, 1\}$ (the set of truth values).

We need a representation language to refer to the objects in our semantic structures. The language of the typed lambda calculus (with its familiar interpretation with respect to standard models) will serve this purpose.

Definition 3.1 *Syntax of typed lambda terms.* Let \mathcal{V}^A be the set of variables of type A . The set Λ of typed λ terms is $\{\mathcal{T}^A\}_{A \in \mathcal{F}}$, where for all $A, B \in \mathcal{F}$:

$$\begin{aligned} \mathcal{T}^A &::= \mathcal{V}^A \mid \mathcal{T}^{B \rightarrow A}(\mathcal{T}^B) \mid (\mathcal{T}^{A \circ B})_0 \mid (\mathcal{T}^{B \circ A})_1 \\ \mathcal{T}^{A \rightarrow B} &::= \lambda \mathcal{V}^A \mathcal{T}^B \quad \mathcal{T}^{A \circ B} ::= \langle \mathcal{T}^A, \mathcal{T}^B \rangle \end{aligned}$$

We now have all the ingredients for presenting term assignment to **LP** sequent proofs. We proceed in two stages: first we present the algorithm for decorating **LP** derivations with intuitionistic term labeling. For Intuitionistic Logic, there is a perfect correspondence between (Natural Deduction) proofs and Λ terms. But not every intuitionistic theorem is **LP** derivable. In the second stage, then, we identify a sublanguage $\Lambda(\text{LP})$ of terms which effectively correspond to the resource-sensitive **LP** derivations.

Definition 3.2 *Term assignment for LP.* Notation: x, y, z for variables, t, u, v for arbitrary terms; $u[t/x]$ for the substitution of term t for variable x in term u . In sequents $x_1 : A_1, \dots, x_n : A_n \Rightarrow t : B$, the antecedent x_i are distinct. For the implication \rightarrow , the rule of use corresponds to functional application, the rule of proof to functional abstraction (λ binding). For \circ , the rule of proof corresponds to pairing, the rule of use to projection. The Cut rule corresponds to substitution.

$$\begin{aligned} \frac{}{x : A \Rightarrow x : A} \text{ (Ax)} \quad \frac{\Gamma \Rightarrow t : A \quad x : A, \Delta \Rightarrow u : B}{\Gamma, \Delta \Rightarrow u[t/x] : B} \text{ (Cut)} \\ \\ \frac{\Gamma, x : A, y : B, \Delta \Rightarrow t : C}{\Gamma, y : B, x : A, \Delta \Rightarrow t : C} \text{ (P)} \\ \\ \frac{\Delta \Rightarrow t : A \quad \Gamma, x : B \Rightarrow u : C}{\Gamma, \Delta, y : A \rightarrow B \Rightarrow u[y(t)/x] : C} (\rightarrow L) \quad \frac{\Gamma, x : A \Rightarrow t : B}{\Gamma \Rightarrow \lambda x.t : A \rightarrow B} (\rightarrow R) \\ \\ \frac{\Gamma \Rightarrow t : A \quad \Delta \Rightarrow u : B}{\Gamma, \Delta \Rightarrow \langle t, u \rangle : A \circ B} (\circ R) \quad \frac{\Gamma, x : A, y : B \Rightarrow t : C}{\Gamma, z : A \circ B \Rightarrow t[(z)_0/x, (z)_1/y] : C} (\circ L) \end{aligned}$$

Unlike intuitionistic resource management, where the structural rules of Contraction and Weakening are freely available, the **LP** regime requires every resource in a proof to be used exactly once. For the implicational fragment, Prop 3.4 indicates how the resource sensitivity of **LP** translates into the syntactic properties of its proof terms, as specified in Def 3.3.²

Definition 3.3 *Let $\Lambda(\text{LP})$ be the largest $\Gamma \subseteq \Lambda$ such that*

- (i) *each subterm of $t \in \Gamma$ contains a free variable*
- (ii) *no subterm of $t \in \Gamma$ contains more than one free occurrence of the same variable*
- (iii) *each occurrence of the λ abstractor in $t \in \Gamma$ binds a variable within its scope*

²For the product, one needs an auxiliary notion specifying what it means for the variables associated with the use of \circ to be used 'exactly once', cf. [Roorda 91]. In Linear Logic, alternative term assignment for the product is available in terms of a construct which directly captures the resource sensitivity of the proof regime: **let** s **be** $x \circ y$ **in** t . See [Troelstra 92].

Proposition 3.4 [van Benthem 87, Buszkowski 87, Wansing 92b]. *Correspondence between LP proofs and $\Lambda(\text{LP})$ terms. Given an LP derivation of a sequent $\sigma = A_1, \dots, A_n \Rightarrow B$ one can find a construction $t^B \in \Lambda(\text{LP})$ of σ , and conversely (where a term $t^B \in \Lambda(\text{LP})$ is called a construction of a sequent $A_1, \dots, A_n \Rightarrow B$ iff t has exactly the free variable occurrences $x_1^{A_1}, \dots, x_n^{A_n}$.)*

IDENTIFYING PROOFS. So far we have been concerned with individual terms, not with relations of equivalence and reducibility between terms. Given the standard interpretation of the Λ term language, the equations (E1) to (E4) of Def 3.5 represent semantic equivalences of certain terms. Read from left (redex) to right (contractum), these equivalences can be seen as valid term reductions. From the Gentzen proof-theoretic perspective, it is natural to look for the operations on proofs that correspond to these term reductions.

Definition 3.5 *Term equations and their proof-theoretic reflexes ([Lambek 93a, Wansing 92b]). (E1) and (E3) correspond to β reduction, (E2) and (E4) to η reduction for function and product types, respectively.*

$$\begin{array}{ll} (E1) & (\lambda x^A . t^B)u = t[u/x] & \text{principal Cut on } A \rightarrow B \\ (E2) & \lambda x^A . (tx)^B = t & \text{non-atomic axiom } A \rightarrow B \\ (E3) & \langle \langle t^A, u^B \rangle_0 = t \quad \langle \langle t, u \rangle_1 = u & \text{principal Cut on } A \circ B \\ (E4) & \langle \langle t^{A \circ B} \rangle_0, \langle t^{A \circ B} \rangle_1 \rangle = t & \text{non-atomic axiom } A \circ B \end{array}$$

The terms for cut-free proofs are in β -normal form: the principal Cut Elimination step replaces a redex by its contractum. Proofs restricted to atomic Axioms yield η -expanded terms. Such proofs can always be simplified by substituting complex Axioms for their unfoldings, yielding η -normal proof terms. The search space for Cut-free proofs is finite. Exhaustive Cut-free search produces the finite number of LP readings, thus providing a proof-theoretic perspective on the Finite Reading Property for LP established in [van Benthem 83].

Example 3.6 *Principal cut: β -conversion. Input:*

$$\frac{\frac{\Gamma, x : B \Rightarrow t : A}{\Gamma \Rightarrow \lambda x.t : A/B} [R/]}{\Delta, \Gamma, \Delta', \Delta'' \Rightarrow v[y(u)/z][\lambda x.t/y] : C} \frac{\frac{\Delta' \Rightarrow u : B \quad \Delta, z : A, \Delta'' \Rightarrow v : C}{\Delta, y : A/B, \Delta', \Delta'' \Rightarrow v[y(u)/z] : C} [L/]}{[Cut]}$$

Output:

$$\frac{\frac{\Delta' \Rightarrow u : B \quad \Gamma, x : B \Rightarrow t : A}{\Gamma, \Delta' \Rightarrow t[u/x] : A} [Cut]}{\Delta, \Gamma, \Delta', \Delta'' \Rightarrow v[t[u/x]/z] : C} \frac{\Delta, z : A, \Delta'' \Rightarrow v : C}{[Cut]}$$

Example 3.7 *Complex axioms: η -conversion.*

$$\frac{\frac{x : B \Rightarrow x : B \quad y : A \Rightarrow y : A}{v : A/B, x : B \Rightarrow y[v(x)/y] : A} [L/]}{v : A/B \Rightarrow \lambda x.y[v(x)/y] : A/B} [R/]}{\sim} \frac{}{v : A/B \Rightarrow v : A/B} [Ax]$$

TERM ASSIGNMENT FOR SYNTACTICALLY MORE DISCRIMINATING SYSTEMS. In moving to the syntactically more discriminating inhabitants of the categorial landscape, we have two options for setting up the term assignment. The primary interest of the working linguist is not so much in the two-way correspondence between terms and proofs, but rather in the one-way computation of a meaning recipe as an automatic spin-off of proof search. From this perspective, one can

be perfectly happy with **LP** term decoration also for the logics with a more developed structure-sensitivity. One relies here on a healthy division of labour between the syntactic and semantic dimensions of the linguistic resources. The role of the uniform $\Lambda(\mathbf{LP})$ term labeling is to capture the composition of signs *qua* semantic objects. Linguistic composition in the form dimension is captured in the term structure over antecedent assumptions (or, alternatively, in terms of a *structural* term labeling discipline for type formulae as discussed in Def ??). As the common denominator of the various calculi in the categorial hierarchy, **LP** can play the role of a general-purpose language of semantic composition. (In LFG, **LP** functions in a similar way as the semantic ‘glue’ language, cf. [Dalrymple e.a. 95].)

In order to accommodate the dualism between syntactic and semantic types, we define a mapping $t : \mathcal{F} \mapsto \mathcal{F}'$ from syntactic to semantic types, which interprets complex types modulo directionality.

$$t(A/B) = t(B \setminus A) = t(B) \rightarrow t(A), \quad t(A \bullet B) = t(A) \circ t(B) \quad (14)$$

The primitive type inventory is a second source of divergence: categorizing signs in their syntactic and semantic dimensions may lead to different choices of atomic types. (For example, both common nouns (n) and verb phrases ($np \setminus s$) may be mapped to the semantic type $e \rightarrow t$ of properties.)

Definition 3.8 *Term assignment for ‘sublinear’ calculi **NL**, **L**, **NLP** using $\Lambda(\mathbf{LP})$ as the language of semantic composition. Structural rules, if any, are neutral with respect to term assignment: they manipulate formulae with their associated term labels.*

$$\begin{array}{c} \text{[Ax]} \frac{}{x : A \Rightarrow x : A} \quad \frac{\Delta \Rightarrow u : A \quad \Gamma[x : A] \Rightarrow t : C}{\Gamma[\Delta] \Rightarrow t[u/x] : C} \text{[Cut]} \\ \\ \text{[R]} \frac{(\Gamma, x : B) \Rightarrow t : A}{\Gamma \Rightarrow \lambda x.t : A/B} \quad \frac{\Delta \Rightarrow t : B \quad \Gamma[x : A] \Rightarrow u : C}{\Gamma[(y : A/B, \Delta)] \Rightarrow u[y(t)/x] : C} \text{[L]} \\ \\ \text{[R]} \frac{(x : B, \Gamma) \Rightarrow t : A}{\Gamma \Rightarrow \lambda x.t : B \setminus A} \quad \frac{\Delta \Rightarrow t : B \quad \Gamma[x : A] \Rightarrow u : C}{\Gamma[(\Delta, y : B \setminus A)] \Rightarrow u[y(t)/x] : C} \text{[L]} \\ \\ \text{[L]} \frac{\Gamma[(x : A, y : B)] \Rightarrow t : C}{\Gamma[z : A \bullet B] \Rightarrow t[(z)_0/x, (z)_1/y] : C} \quad \frac{\Gamma \Rightarrow t : A \quad \Delta \Rightarrow u : B}{(\Gamma, \Delta) \Rightarrow \langle t, u \rangle : A \bullet B} \text{[R]} \end{array}$$

The alternative for the dualistic view is to equip the various inhabitants of the categorial landscape with more structured semantic term languages which directly reflect the syntactic resource management regime of the logics in question. In [Buszkowski 87, Wansing 92b, Hepple 94] one finds a term language which distinguishes left- and right oriented forms of abstraction λ^l, λ^r and application. These allow for a refinement of the term restrictions characterizing the $\Lambda(\mathbf{L})$ fragment and the two-way correspondence between term constructions and proofs: in the case of **L** the left (right) abstractors bind the leftmost (rightmost) free variable in their scope. In a similar vein, one could look for a structural characterization of the non-associativity of **NL**.

As long as the interpretation of the types is given in terms of function spaces and Cartesian products, the distinctions between left/right abstraction/application remain purely syntactic. For a more ambitious programme, see [Abrusci 96], who proposes a refinement of the notion ‘meaning of proofs’ in the context of a generalization of the coherence semantics for Linear Logic. One considers *bimodules* on coherent spaces and refines the class of linear functions into left-linear and right-linear functions. Interpreting A on the coherent space X and B on the coherent space Y , $D_{A \setminus B}$ (resp. $D_{B/A}$) is the coherent space of all the left-linear (resp. right-linear) functions from X to Y .

NATURAL DEDUCTION. For the display of sample derivations in the following section, we will continue to use the handy natural deduction format, which is presented below in its term-annotated form.

Definition 3.9 *Term assignment: (sequent-style) Natural Deduction.* Notation: $\Gamma \vdash t : A$ for a deduction of the formula A decorated with term t from a structured configuration of undischarged term-decorated assumptions Γ .

$$\begin{array}{c}
 x : A \vdash x : A \\
 \\
 \begin{array}{c}
 \text{[I]} \frac{(\Gamma, x : B) \vdash t : A}{\Gamma \vdash \lambda x.t : A/B} \qquad \frac{\Gamma \vdash t : A/B \quad \Delta \vdash u : B}{(\Gamma, \Delta) \vdash t(u) : A} \text{[E]} \\
 \\
 \text{[\backslash I]} \frac{(x : B, \Gamma) \vdash t : A}{\Gamma \vdash \lambda x.t : B \backslash A} \qquad \frac{\Gamma \vdash u : B \quad \Delta \vdash t : B \backslash A}{(\Gamma, \Delta) \vdash t(u) : A} \text{[\backslash E]} \\
 \\
 \text{[\bullet I]} \frac{\Gamma \vdash t : A \quad \Delta \vdash u : B}{(\Gamma, \Delta) \vdash \langle t, u \rangle : A \bullet B} \qquad \frac{\Delta \vdash u : A \bullet B \quad \Gamma[(x : A, y : B)] \vdash t : C}{\Gamma[\Delta] \vdash t[(u)_0/x, (u)_1/y] : C} \text{[\bullet E]}
 \end{array}
 \end{array}$$

3.2 Natural language interpretation: the deductive view

For an assessment of categorial type logics in the context of Montague’s Universal Grammar program, it is instructive to compare the type-logical *deductive* view on the composition of linguistic meaning with the standard Montagovian *rule-to-rule* philosophy as discussed in Chapter One. The rule-to-rule view on the syntax-semantics interface characterizes syntax in terms of a collection of syntactic rules (or rule schemata); for every syntactic rule, there is a corresponding semantic rule, specifying how the meaning of the whole is put together in terms of the meaning of the parts and the way they are put together. Apart from the homomorphism requirement for the syntactic and semantic algebras, compositionality, in its rule-to-rule implementation, does not impose any principled restrictions on exactly what operations in the semantic algebra one wants to line up with the syntactic algebra: the correlation between syntactic and semantic rules/operations can be entirely stipulative.

The type logical approach, as we have seen in §2 eliminates ‘syntax’ as a component of primitive rules. Instead of syntactic rules, one finds theorems — deductive consequences derived from the interpretation of the type-constructors. In the absence of syntactic rules, there can be no rule-to-rule stipulated assignment of meaning to derivations: rather, every theorem has to derive its meaning from its proof, again purely in terms of the semantic action of the type-constructors under the Curry-Howard correspondence.

Example 3.10 *Argument lowering ([Partee & Rooth 83]): the lexical type assignment for the verb ‘needs’, $(np \backslash s) / ((s/np) \backslash s)$, can be lowered to $(np \backslash s) / np$. As discussed in §2, the principle is generally valid in the pure residuation logic NL.*

$$\begin{array}{c}
 \frac{\text{needs}}{\text{needs} : (np \backslash s) / ((s/np) \backslash s)} \quad \frac{\frac{x_1 : s/np \quad x_0 : np}{x_1(x_0) : s} /E}{\lambda x_1.x_1(x_0) : (s/np) \backslash s} \backslash I \\
 \frac{\text{needs}(\lambda x_1.x_1(x_0)) : np \backslash s}{\lambda x_0.\text{needs}(\lambda x_1.x_1(x_0)) : (np \backslash s) / np} /I
 \end{array}$$

DERIVATIONAL AMBIGUITY: PROOFS AND READINGS. The rule-to-rule implementation of compositionality requires there to be a unique meaning assignment for every syntactic rule. If one would like to associate different semantic effects with what looks like one and the same syntactic rule, one has to introduce diacritics in the syntax in order to keep the homomorphism requirement intact. In contrast, for the type-logical approach meaning resides in the *proof*, not in the type-change theorem that labels the conclusion of a proof. Different ways of proving one and the same goal sequent may, or may not, result in different readings.

Example 3.11 *As an example of derivational ambiguity, we consider the type-shifting principle known as Argument Raising ([Partee & Rooth 83]). The derivations below represent two semantically distinct \mathbf{L} proofs of the theorem $(np \setminus s) / np \Rightarrow ((s / (np \setminus s)) \setminus s) / ((s / np) \setminus s)$, turning a simple first-order transitive verb into a third-order functor taking second-order generalized quantifier type arguments, encoding the subject wide scope reading (\dagger) and object wide scope (\ddagger) reading, respectively.*

$$\begin{array}{c}
 \dagger \\
 \frac{\frac{\frac{\mathbf{tv}}{\mathbf{tv} : (np \setminus s) / np \quad x_1 : np}}{x_0 : np \quad \mathbf{tv}(x_1) : np \setminus s} / E}{\mathbf{tv}(x_1)(x_0) : s} / I \quad \frac{}{x_2 : (s / np) \setminus s} / E}{\lambda x_1. \mathbf{tv}(x_1)(x_0) : s / np} / E \\
 \frac{\frac{\frac{\frac{x_2(\lambda x_1. \mathbf{tv}(x_1)(x_0)) : s}{\lambda x_0. x_2(\lambda x_1. \mathbf{tv}(x_1)(x_0)) : np \setminus s} \setminus I}{x_3 : s / (np \setminus s)} / E}{x_3(\lambda x_0. x_2(\lambda x_1. \mathbf{tv}(x_1)(x_0))) : s} / I \quad \frac{}{x_2 : (s / np) \setminus s} / E}{\lambda x_3. x_3(\lambda x_0. x_2(\lambda x_1. \mathbf{tv}(x_1)(x_0))) : (s / (np \setminus s)) \setminus s} / I \\
 \frac{}{\lambda x_2 \lambda x_3. x_3(\lambda x_0. x_2(\lambda x_1. \mathbf{tv}(x_1)(x_0))) : ((s / (np \setminus s)) \setminus s) / ((s / np) \setminus s)} / I
 \end{array}$$

$$\begin{array}{c}
 \ddagger \\
 \frac{\frac{\frac{\mathbf{tv}}{\mathbf{tv} : (np \setminus s) / np \quad x_1 : np}}{x_0 : np \quad \mathbf{tv}(x_1) : np \setminus s} / E}{\mathbf{tv}(x_1)(x_0) : s} / I \quad \frac{}{x_3 : (s / np) \setminus s} / E}{\frac{\frac{\frac{x_2(\lambda x_0. \mathbf{tv}(x_1)(x_0)) : s}{\lambda x_0. \mathbf{tv}(x_1)(x_0) : np \setminus s} \setminus I}{x_2 : s / (np \setminus s)} / E}{\lambda x_1. x_2(\lambda x_0. \mathbf{tv}(x_1)(x_0)) : s / np} / I \quad \frac{}{x_3 : (s / np) \setminus s} / E}{x_3(\lambda x_1. x_2(\lambda x_0. \mathbf{tv}(x_1)(x_0))) : s} / I \quad \frac{}{x_3 : (s / np) \setminus s} / E}{\lambda x_2. x_3(\lambda x_1. x_2(\lambda x_0. \mathbf{tv}(x_1)(x_0))) : (s / (np \setminus s)) \setminus s} / I \\
 \frac{}{\lambda x_3 \lambda x_2. x_3(\lambda x_1. x_2(\lambda x_0. \mathbf{tv}(x_1)(x_0))) : ((s / (np \setminus s)) \setminus s) / ((s / np) \setminus s)} / I
 \end{array}$$

LEXICAL VERSUS DERIVATIONAL SEMANTICS. The derivational semantics of a sequent $\Gamma \Rightarrow t : A$ gives a meaning recipe t in terms of free variables x_i for the antecedent assumptions A_i in Γ , the ‘parameters’ of the recipe. In the actual computation of the meaning of a natural language expression, we substitute the *lexical* meanings of the words constituting the expression for these variables. For the logically more exciting part of the vocabulary, this will involve the substitution of a compound λ term representing the lexical meaning for a parameter in the proof term. The strict division of labour between the role assigned to derivational and lexical semantics realizes a fully modular implementation of compositionality, which has a number of pleasant consequences: on the level of individual lexical items, lexical semantics can overcome the expressive limitations of the resource-conscious derivational component; on a more global level, one can interface the neutral derivational semantics with one’s favourite semantic theory via an appropriate category-to-type mapping and lexical meaning assignment. We illustrate these two aspects in turn.

NON-LINEAR MEANING RECIPES. We saw that the resource-sensitive **LP** terms have the property that every assumption is used exactly once: the lambda operator binds exactly one variable occurrence. Natural language semantics, in a variety of constructions, requires the identification of variables. Assigning multiple-bind terms to the relevant classes of lexical items one can realize variable-sharing while maintaining the resource-sensitivity of derivational semantics.

Example 3.12 ‘Everyone loves himself’. Proof term and substitution of lexical recipes. (Notice that reductions after lexical substitution can destructively affect the proof term, in the sense that the original proof term becomes irrecoverable after the ‘lexical’ β conversions.)

$$\begin{array}{c}
\text{himself}_{((np \setminus s)/np) \setminus (np \setminus s)} := \lambda x \lambda y. x(y)(y) \quad \text{everyone}_{s/(np \setminus s)} := \lambda x \forall y (\text{person}(y) \Rightarrow x(y)) \\
\\
\frac{\frac{\frac{\text{loves}}{\text{loves} : (np \setminus s)/np \quad x_2 : np}}{x_1 : np} /E}{\text{loves}(x_2) : np \setminus s} \setminus E}{\frac{\text{loves}(x_2)(x_1) : s}{\lambda x_1. \text{loves}(x_2)(x_1) : np \setminus s} \setminus I} /I}{\frac{\lambda x_2 \lambda x_1. \text{loves}(x_2)(x_1) : (np \setminus s)/np}{x_0 : np} /I \quad \frac{\text{himself}}{\text{himself} : ((np \setminus s)/np) \setminus (np \setminus s)} \setminus E} \setminus E}{\text{himself}(\lambda x_2 \lambda x_1. \text{loves}(x_2)(x_1)) : np \setminus s} \setminus E} \setminus E}{\frac{\text{everyone}}{\text{everyone} : s/(np \setminus s)} \quad \frac{\text{himself}(\lambda x_2 \lambda x_1. \text{loves}(x_2)(x_1))(x_0) : s}{\lambda x_0. \text{himself}(\lambda x_2 \lambda x_1. \text{loves}(x_2)(x_1))(x_0) : np \setminus s} \setminus I} /E} \setminus E} \\
\text{everyone}(\lambda x_0. \text{himself}(\lambda x_2 \lambda x_1. \text{loves}(x_2)(x_1))(x_0)) : s \\
\text{everyone}(\lambda x_0. \text{himself}(\lambda x_2 \lambda x_1. \text{loves}(x_2)(x_1))(x_0)) \rightarrow_{\beta} \forall y. \text{person}(y) \Rightarrow \text{love}(y)(y)
\end{array}$$

DERIVATIONAL SEMANTICS: PORTABILITY. The proof terms associated with categorial derivations relate structural composition in a systematic way to the composition of meaning. The derivational semantics is fully neutral with respect to the particular ‘theory of natural language semantics’ one wants to plug in: an attractive design property of the type-logical architecture when it comes to portability. An illustration can be found in [Muskens xx], who proposes a type-logical emulation of Discourse Representation Theory (cf. Chapter Three) driven by a categorial proof engine.

To obtain the combination, one starts from an appropriate primitive type inventory for dynamic semantics: e and t as before, plus s (program states) and d (‘pigeon-holes’ for discourse referents). The category to type map is set up in such a way that the syntactic categories get an interpretation in the appropriate semantic domains: $t(s) = t(txt) = s \rightarrow s \rightarrow t$, $t(np) = d$, $t(n) = d \rightarrow (s \rightarrow (s \rightarrow t))$. Lexical recipes for a tiny corner of the DRT lexicon are given in (15).

a^n	$(s/(np \setminus s))/n$	$\lambda P \lambda Q. [u_n \mid]; P(u_n); Q(u_n)$	
man, woman	n	$\lambda v. [\mid \text{MAN } v], \lambda v. [\mid \text{WOMAN } v]$	
loves, hates	$(np \setminus s)/np$	$\lambda v \lambda w. [\mid v \text{ LOVES } w], \lambda v \lambda w. [\mid v \text{ HATES } w]$	(15)
he ⁿ , she ⁿ	$s/(np \setminus s)$	$\lambda P. P(u_n)$	
him ⁿ , her ⁿ	$(s/np) \setminus s$	$\lambda P. P(u_n)$	
.	$s \setminus (txt/s), txt \setminus (txt/s)$	$\lambda p \lambda q. p; q$	

The reader will be able to verify in Ex 3.13 that the little discourse (a) is associated with proof term (b) which reduces to the discourse representation structure (c), using some notational sugaring, detailed below as far as relevant.

Example 3.13 *Type-driven composition of DRS's ([Muskens xx]). Notational abbreviations:*

$$\begin{aligned} \phi; \psi &= \lambda i \lambda j \exists k. \phi i k \wedge \psi i k && \text{(dynamic discourse composition)} \\ [u_1 \dots u_n \mid \gamma_1, \dots, \gamma_n] &= \lambda i \lambda j. i [u_1 \dots u_n] j \wedge \gamma_1(j) \wedge \dots \wedge \gamma_n(j) && \text{(box with conditions } \gamma_i) \\ [\bar{u} \mid \bar{\gamma}]; [\bar{u}' \mid \bar{\gamma}'] &\rightsquigarrow [\bar{u} \bar{u}' \mid \bar{\gamma} \bar{\gamma}'] && \text{(merging boxes, provided the } \bar{u}' \text{ do not occur in any of the } \bar{\gamma}) \end{aligned}$$

- (a) $A^1 \text{ man loves } a^2 \text{ woman. She}_2 \text{ hates him}_1$
- (b) $\cdot (a(\text{man})(\lambda x. a(\text{woman})(\lambda y. \text{loves}(y)(x)))) (\text{she}(\lambda v. \text{him}(\lambda w. \text{hates}(w)(v))))$
- (c) $[u_1, u_2 \mid \text{MAN } u_1, \text{WOMAN } u_2, u_1 \text{ LOVES } u_2, u_2 \text{ HATES } u_1]$

Discussion: quantifier scope ambiguities

We close this section with a discussion of scope ambiguities involving generalized quantifier expressions: these phenomena nicely illustrate the tension between the composition of form and meaning, and the different strategies for resolving these tensions.

Consider generalized quantifier expressions like ‘someone’, ‘everybody’. From the perspective of **LP**, we can study their semantic contribution via a standard Fregean type assignment $(e \rightarrow t) \rightarrow t$, with lexical recipes $\lambda x. \exists y[x(y)]$, $\lambda x. \forall y[x(y)]$. The **LP** notion of derivability, of course, is too crude to offer a unified deductive account of semantics in conjunction with syntax. Suppose we want to refine the **LP** type $(e \rightarrow t) \rightarrow t$ to take syntactic fine-structure into account. Within **L**, one can find two directional realizations compatible with the fact that generalized quantifiers occupy the positions of ordinary (proper noun) noun phrases: $s/(np \setminus s)$ and $(s/np) \setminus s$. But imposing order sensitivity in the type-assignment already causes the loss of scope readings one wants to preserve. Compare ‘peripheral’ versus ‘medial’ occurrences of generalized quantifiers. Given a ‘direct object’ assignment $(s/np) \setminus s$ to ‘someone’, both the (a) and (a’) readings are **L**-derivable. Given a ‘subject’ assignment $s/(np \setminus s)$ the (b) reading is not derivable: in **L** one only derives the narrow scope reading (b’).

$$\begin{aligned} (a) \quad \text{Suzy thinks Mary loves someone} &\rightsquigarrow \text{someone}(\lambda x. \text{thinks}(\text{loves}(x)(\mathbf{m}))(s)) \\ (a') &\rightsquigarrow \text{thinks}(\text{someone}(\lambda x. \text{loves}(x)(\mathbf{m}))(s)) \\ (b) \quad \text{Suzy thinks someone loves Mary} &\rightsquigarrow \text{someone}(\lambda x. \text{thinks}(\text{loves}(\mathbf{m})(x))(s)) \\ (b') &\rightsquigarrow \text{thinks}(\text{someone}(\lambda x. \text{loves}(\mathbf{m})(x))(s)) \end{aligned} \tag{16}$$

The diagnosis of the problem is easy in the light of §2: the (b) reading would require the generalized quantifier expression to enter into structural composition with a *discontinuous* configuration of resources: such syntactic behaviour is beyond the expressivity of the **(N)L** connectives:

$$\boxed{\text{Suzy thinks}} \text{ someone } \boxed{\text{loves Mary}}$$

We compare two strategies to resolve this problem: (i) in the *rule-based* approach, one postulates type-change axiom schemata to regain the lost readings, (ii) in the *deductive* approach: one enriches the vocabulary of connectives with logical constants such that these axiom schemata become derivable theorems. Flexible Montague Grammar ([Hendriks 93]), and the closely related polymorphic approach of [Emms 93b] (to be taken up in §??) are representatives of (i). The deductive alternative has been developed in [Moortgat 91, Morrill 95a, Carpenter 96].

FLEXIBLE MONTAGUE GRAMMAR. Hendriks’ proposal is formulated as a flexible version of Montague Grammar (FMG). For an assessment in the Montagovian context we refer to Chapter One. Our objective here is to give a type-logical reconstruction of the essential ideas. Syntactically, FMG is restricted to combine phrases by means of function application rule schemata. In order to accommodate quantificational scope ambiguities, the category-to-type mapping is relaxed to a relation rather than a function: a given syntactic type is associated with a set of semantic types. The semantic types are not unrelated: from a generator type an infinite number of semantic types (and

the associated meaning recipes) are derived via the type-shifting rule schemata of Value Raising (VR), Argument Lowering (AL), and Argument Raising (AR).

Let us identify the pure application syntax of FMG as **NL**, and try to pinpoint exactly where the type-shifting schemata give a surplus inferential capacity. As we have seen in §2, Value Raising and Argument Lowering are universally valid already in the pure residuation logic **NL**: they reflect the monotonicity properties of the implicational type constructors. Argument Raising, as a semantic type-shifting rule, is schematically characterized in (17) (where $\vec{A} \rightarrow B$ abbreviates $A_1 \rightarrow \dots \rightarrow A_n \rightarrow B$, and similarly for \vec{x}).

$$(AR) \quad \begin{array}{l} \vec{A} \rightarrow B \rightarrow \vec{C} \rightarrow D \quad \rightarrow \quad \vec{A} \rightarrow ((B \rightarrow D) \rightarrow D) \rightarrow \vec{C} \rightarrow D \\ t \quad \rightarrow \quad \lambda \vec{x}_{\vec{A}} \lambda w_{(B \rightarrow D) \rightarrow D} \lambda \vec{y}_{\vec{C}} . w(\lambda z_B . t(\vec{x})(z)(\vec{y})) \end{array} \quad (17)$$

Directional realizations of this schema are *not* generally valid. We saw two special cases in our Example 3.11: these happened to be derivable, in the associative setting of **L**, for generalized quantifiers occupying peripheral positions in their scopal domain. But what we would like to have in full generality is the possibility of having a generalized quantifier phrase at any np position, exerting its binding force at any s level of embedding.

As an illustration for the FMG type-shifting approach, take the sentence ‘Kazimierz thinks someone left’. In (18) we list the necessary steps producing the wide scope reading for ‘someone’. We give both the semantic shifts — abbreviating $A \rightarrow B$ as (AB) — and their directional counterpart. The (AR) transition for ‘left’, with the generalized quantifier variable x_1 in head position, is the critical one that cannot be obtained as a pure **NL** proof term. Combining the words (in their shifted types) by means of functional application produces the desired reading.

$$\begin{array}{ll} \text{thinks} & (np \setminus s) / s \Rightarrow (np \setminus s) / ((s/s) \setminus s) \\ & (t(et)) \Rightarrow_{AR} ((tt)t)(et) \\ & \text{thinks} \Rightarrow \lambda x_2 \lambda x_0 . x_2(\lambda x_1 . \text{thinks}(x_1)(x_0)) \\ \text{left} & np \setminus s \Rightarrow np \setminus ((s/s) \setminus s) \\ & (et) \Rightarrow_{VR} (e((tt)t)) \\ & \text{left} \Rightarrow \lambda x_1 \lambda x_0 . x_0(\text{left}(x_1)) \quad (= \text{left}') \\ & \\ & np \setminus ((s/s) \setminus s) \Rightarrow (s / (np \setminus s)) \setminus ((s/s) \setminus s) \\ & (e((tt)t)) \Rightarrow_{AR} ((et)t)((tt)t) \\ & \text{left}' \Rightarrow \lambda x_2 \lambda x_0 . x_2(\lambda x_1 . x_0(\text{left}(x_1))) \end{array} \quad (18)$$

A CONNECTIVE FOR BINDING. The deductive alternative is to investigate the theoretical space provided by the Lambek landscape in order to identify within this space a logical constant which renders the critical AR cases (the cases beyond the reach of **(NL)** derivable.

Definition 3.14 *In situ binding* $q(A, B, C)$ ([Moortgat 91]). *Use of a formula* $q(A, B, C)$ *binds a variable* x *of type* A , *where the resource* A *is substituted for (takes the place of)* $q(A, B, C)$ *in the binding domain* B . *Using* $q(A, B, C)$ *turns the binding domain* B *into* C . *In the generalized quantifier case we have typing* $q(np, s, s)$ *where it happens that* $B = C = s$. *For the semantic term decoration of the rule of use* $[qL]$, *assume* $t(q(A, B, C)) = (t(A) \rightarrow t(B)) \rightarrow t(C)$.

$$\frac{\Delta[x : A] \Rightarrow t : B \quad \Gamma[y : C] \Rightarrow u : D}{\Gamma[\Delta[z : q(A, B, C)]] \Rightarrow u[z(\lambda x.t)/y] : D} \quad (qL)$$

Example 3.15 *Direct cut-free proof search for ‘Kazimierz thinks someone left’, with wide scope ‘someone’. (Compare: the FMG strategy of (18).)*

$$\frac{\frac{\frac{np \Rightarrow np \quad s \Rightarrow s}{np, np \setminus s \Rightarrow s} (\setminus L) \quad \frac{np \Rightarrow np \quad s \Rightarrow s}{np, np \setminus s \Rightarrow s} (\setminus L)}{\frac{np, (np \setminus s) / s, \boxed{x : np}, np \setminus s \Rightarrow \boxed{u : s}}{np, (np \setminus s) / s, \boxed{\text{someone}:q(np, s, s)}, np \setminus s \Rightarrow \boxed{y[\text{someone}(\lambda x.u)/y] : s}} (\setminus L)}{\boxed{y : s} \Rightarrow y : s} \quad (qL)$$

$$u = \mathit{thinks}(\mathit{left}(x))(k), \quad y[\mathit{someone}(\lambda x.u)/y] = \mathit{someone}(\lambda x.\mathit{thinks}(\mathit{left}(x))(k))$$

Carpenter [Carpenter 94, Carpenter 96] offers an in-depth discussion of the empirical range of the binding connective as compared with competing approaches to quantification, and an extension with a treatment of plurals. Notice finally the different ‘heuristic’ qualities of the connective-based and the rule-based type-shifting alternatives. The type-shifting approach is specifically designed to handle the semantics of quantificational phenomena and obtain minimal type assignment. The deductive approach introduces a *connective*, i.e. a fully general operation on types that cannot have a construction-specific limited range of application. Support for the generality of a connective for *in situ* binding can be found in the analyses of Pied-Piping ([Morrill 95a]), or *more ... than* comparative subdeletion ([Hendriks 95]).

We close this discussion with some open questions. With the $[qL]$ inference, we have given a rule of *use* — what about the rule of proof for the *in situ* binder? Also, the q connective was presented as a primitive connective, whereas the term assignment $z(\lambda x.t)$ shows the interaction of two implications— could we decompose the q connective into more elementary logical constants? In the context of the simple type logics we are discussing here, these questions must remain unanswered. In section §4.1, multimodal type logics will be introduced which provide the tools to tackle these issues in a principled way.

4 Grammatical composition: multimodal systems

In the present section we generalize the multiplicative vocabulary in a number of directions. The generalizations do not affect the overall model-theoretic or proof-theoretic properties of the categorial architecture in any essential sense. But they increase the linguistic sophistication in such a way that the limitations of the simple systems discussed in §2.3 are overcome.

In §4.1, simple type logics are put together into a mixed, *multimodal* system where distinct notions of grammatical composition coexist and communicate. The multimodal style of reasoning was developed in the work of Oehrle, Morrill, Hepple and the author, cf. [Moortgat & Morrill 91, Moortgat & Oehrle 93, Moortgat & Oehrle 94, Hepple 94]. This development reintroduces in the type-logical discussion the theme of the ‘multidimensionality’ of grammatical composition that had been dealt with on a more philosophical level in earlier work such as [Oehrle 88, Bach 84]. Another antecedent is [Dowty 91], who distinguishes composition modes with different degrees of coherence.

In §4.2, the binary vocabulary is extended with a language of *unary* multiplicatives. The unary connectives play the role of *control* devices, with respect to both the static aspects of linguistic structure, and the dynamic aspects of putting this structure together. Unary operations entered the type-logical discussion in [Morrill 90a], who provides an analysis of semantic domains of intensionality in terms of a \Box operator. The unary vocabulary soon found a variety of applications, including the syntactic domain modalities of [Hepple 90], the ‘structural modalities’ of [Barry e.a. 91], and the ‘bracket’ operators of [Morrill 95a]. Our treatment below systematizes and refines these earlier proposals.

As indicated in §1, the developments to be discussed here represent the categorial digestion of a number of themes in the field of Linear Logic and related substructural systems, and of Gabbay’s general program for combining logics. The collection *Substructural Logics* [Došen & Schröder-Heister 93] and [Gabbay 94] offer useful background reading for these lines of research.

4.1 Mixed inference: the modes of composition

In §2 the type-forming connectives $/, \bullet, \backslash$ were interpreted in terms of a *single* notion of linguistic composition. In moving to a *multimodal* architecture the objective is to combine the virtues of

the individual logics we have discussed so far, and to exploit new forms of grammatical inference arising from their communication. In merging different logics into a mixed system, we have to take care that their individual resource management properties are left intact. This can be done by relativizing linguistic composition to specific resource management *modes*. But also, we want the inferential capacity of the combined logic to be more than the sum of the parts. The extra expressivity comes from *interaction postulates* that hold when different modes are in construction with one another. The interaction postulates can apply in full generality, or can themselves be intrinsically controlled by exploiting mode distinctions, or by composition of modes.

On the syntactic level, the category formulae for the multimodal language are defined inductively on the basis of a set of category atoms \mathcal{A} and a set of indices I . We refer to the $i \in I$ as *composition modes*, or modes for short.

$$\mathcal{F} ::= \mathcal{A} \mid \mathcal{F}/_i\mathcal{F} \mid \mathcal{F} \bullet_i \mathcal{F} \mid \mathcal{F} \setminus_i \mathcal{F} \quad (19)$$

The interpretation for the mixed language is a straightforward generalisation of the semantics for the simple systems. Rather than interpret the multiplicatives in terms of *one* privileged notion of linguistic composition, we put together different forms of linguistic composition and interpret in multimodal frames $\langle W, \{R_i^3\}_{i \in I} \rangle$. The valuation v respects the structure of the complex formulae in the familiar way, interpreting each of the modes $i \in I$ in terms of its own composition relation R_i . The basic residuation laws (20) are *relativized* with respect to the composition modes.

Definition 4.1 *Interpretation in multimodal frames* $\langle W, \{R_i^3\}_{i \in I} \rangle$.

$$\begin{aligned} v(A \bullet_i B) &= \{x \mid \exists y \exists z [R_i x y z \ \& \ y \in v(A) \ \& \ z \in v(B)]\} \\ v(C/_i B) &= \{y \mid \forall x \forall z [(R_i x y z \ \& \ z \in v(B)) \Rightarrow x \in v(C)]\} \\ v(A \setminus_i C) &= \{z \mid \forall x \forall y [(R_i x y z \ \& \ y \in v(A)) \Rightarrow x \in v(C)]\} \end{aligned}$$

$$A \rightarrow C/_i B \quad \text{iff} \quad A \bullet_i B \rightarrow C \quad \text{iff} \quad B \rightarrow A \setminus_i C \quad (20)$$

In sequent presentation, each residuated family of multiplicatives $\{/_i, \bullet_i, \setminus_i\}$ has a matching structural connective $(\cdot, \cdot)^i$. Logical rules insist that use and proof of connectives respect the resource management modes. The explicit construction of the antecedent database in terms of structural connectives derives directly from Belnap's [Belnap 82] work on Display Logic, where it serves the same purpose as it does here, viz. to combine logics with different resource management regimes. In [Kracht 93, Wansing 92a] one finds recent applications in the context of modal logic. More recently, the same idea has been introduced in Linear Logic in [Girard 93].

Definition 4.2 *Multimodal Gentzen calculus: logical rules. Structure terms* $S ::= \mathcal{F} \mid (S, S)^i$.

$$\begin{aligned} [R/_i] \frac{(\Gamma, B)^i \Rightarrow A}{\Gamma \Rightarrow A/_i B} & \quad \frac{\Gamma \Rightarrow B \quad \Delta[A] \Rightarrow C}{\Delta[(A/_i B, \Gamma)^i] \Rightarrow C} [L/_i] \\ [R \setminus_i] \frac{(B, \Gamma)^i \Rightarrow A}{\Gamma \Rightarrow B \setminus_i A} & \quad \frac{\Gamma \Rightarrow B \quad \Delta[A] \Rightarrow C}{\Delta[(\Gamma, B \setminus_i A)^i] \Rightarrow C} [L \setminus_i] \\ [L \bullet_i] \frac{\Gamma[(A, B)^i] \Rightarrow C}{\Gamma[A \bullet_i B] \Rightarrow C} & \quad \frac{\Gamma \Rightarrow A \quad \Delta \Rightarrow B}{(\Gamma, \Delta)^i \Rightarrow A \bullet_i B} [R \bullet_i] \end{aligned}$$

Notice that the mode specification can keep apart distinct forms of grammatical composition even if they have *the same* resource management properties. The dependency calculus of [Moortgat & Morrill 91] provides an example. By splitting up the product \bullet in a left-headed \bullet_l and a right-headed \bullet_r , these authors introduce a dimension of *dependency* structure next to the dimensions of precedence and dominance. The dependency products could both be non-associative operators; still, with the mode specification we would be able to distinguish left-headed structures from right-headed ones. Linguistic motivation for the dependency dimension can be found in [Barry 91, Barry & Pickering 90].

In addition to the residuation inferences (the fixed ‘logical’ component for all modes), we can now have mode-specific structural options. For a commutative mode c , for example, we would have the structural postulates (structural rules, in the Gentzen style) below, together with the matching frame constraint for the composition relation interpreting \bullet_c : $(\forall x, y, z \in W) R_cxyz \Rightarrow R_cxzy$. (Where there is an established notation, such as \otimes in this case, we will often use the familiar symbol instead of the ‘official’ \bullet_c notation. Also, we will sometimes abuse notation in using the product symbols themselves as mode indices in the Gentzen term language.)

$$A \bullet_c B \longleftrightarrow B \bullet_c A \quad \frac{\Gamma[(\Delta_2, \Delta_1)^c] \Rightarrow A}{\Gamma[(\Delta_1, \Delta_2)^c] \Rightarrow A} [\text{P}] \quad (21)$$

It is straightforward to extend the completeness results of §2 to the multimodal architecture, cf [Kurtonina 95] for discussion. Semantic annotation of the multimodal derivations with λ term meaning recipes is implemented in exactly the same way as for the unimodal systems.

MULTIMODAL COMMUNICATION. What we have done so far is simply put together the individual systems discussed before in isolation. This is enough to gain combined access to the inferential capacities of the component logics, and one avoids the unpleasant collapse into the least discriminating logic that results from putting together theorems from different simple logics without taking into account the mode distinctions, cf. our discussion in §2. But as things are, the borders between the constituting logics in our multimodal setting are still hermetically closed. Communication between composition relations R_i and R_j can be established in two ways.

Inclusion postulates. Postulates $A \bullet_i B \rightarrow A \bullet_j B$, with corresponding frame conditions $(\forall xyz \in W) R_ixyz \Rightarrow R_jxyz$, impose a ‘specificity’ order on composition modes i, j .

Interaction postulates. Postulates ‘mixing’ distinct modes i, j allow for the statement of distributivity principles regulating the communication between composition modes R_i, R_j . One obtains constrained multimodal forms of the resource management postulates of §2.

INCLUSION PRINCIPLES. One can develop different perspectives on inclusion principles depending on the interpretation one has in mind for the ordering of the composition relations R_i, R_j involved. A natural candidate would be an ordering in terms of the information they provide about the structure of the linguistic resources. From this perspective, the non-commutative product \bullet would count as more informative than the commutative product \otimes , since the former but not the latter is sensitive to the linear order of the resources. In terms of frame conditions, one imposes the constraint $R_\bullet xyz \Rightarrow R_\otimes xyz$, corresponding to the postulate $A \bullet B \rightarrow A \otimes B$. This perspective is taken in general terms in [Moortgat & Oehle 93], where two products \bullet_i and \bullet_j are related by an inclusion principle $A \bullet_i B \rightarrow A \bullet_j B$ if the latter has greater freedom of resource management than the former. The opposite view is taken in [Hepple 95], where one finds a systematic reversal of the derivability arrows in the inclusion principles, e.g. $A \otimes B \rightarrow A \bullet B$. In [Kurtonina 95] it is shown that from the frame semantics point of view the two perspectives can be equally well accommodated: they reflect the choice for a ‘conjunctive’ versus ‘disjunctive’ reading of the commutative product.

INTERACTION PRINCIPLES. Among the multimodal interaction principles, we distinguish cases of weak and strong distributivity. The weak distributivity principles do not affect the multiplicity of the linguistic resources. They allow for the realization of mixed associativity or commutativity laws as the multimodal counterparts of the unimodal versions discussed above. Interaction principles of the strong distributivity type duplicate resources, thus giving access to mode-restricted forms of Contraction.

WEAK DISTRIBUTIVITY. Consider first interaction of the weak distributivity type. Def 4.3 states principles of *mixed* associativity and commutativity. Instead of the global associativity and com-

mutativity options characterizing **L**, **NLP**, **LP**, these principles realize constrained forms of associativity/commutativity, restricted to the situation where modes i and j are in construction. (Symmetric duals can be added with the i mode distributing from the right, and one can split up the two-directional inferences in their one-directional components, if so required.)

Definition 4.3 *Mixed Associativity (MA), Mixed Commutativity (MP). Structural postulates, frame constraints, Gentzen rules.*

$$\begin{aligned} MP : \quad & A \bullet_i (B \bullet_j C) \longleftrightarrow B \bullet_j (A \bullet_i C) \quad \exists t(R_iuxt \ \& \ R_jtyz) \Leftrightarrow \exists t'(R_juyt' \ \& \ R_it'xz) \\ MA : \quad & A \bullet_i (B \bullet_j C) \longleftrightarrow (A \bullet_i B) \bullet_j C \quad \exists t(R_iuxt \ \& \ R_jtyz) \Leftrightarrow \exists t'(R_jut'z \ \& \ R_it'xy) \end{aligned}$$

$$\frac{\Gamma[(\Delta_2, (\Delta_1, \Delta_3)^i)^j] \Rightarrow A}{\Gamma[(\Delta_1, (\Delta_2, \Delta_3)^j)^i] \Rightarrow A} [MP] \quad \frac{\Gamma[(\Delta_1, \Delta_2)^i, \Delta_3)^j] \Rightarrow A}{\Gamma[(\Delta_1, (\Delta_2, \Delta_3)^j)^i] \Rightarrow A} [MA]$$

For linguistic application of these general postulates, we turn to discontinuous dependencies. In the work of authors such as [Bach 84, Pollard 84, Jacobson 87], it has been argued that the discontinuous mode of combination ('wrapping') should be treated as a grammatical operation *sui generis*, rather than simulated in terms of the regular 'concatenation' mode. In the type-logical setting one can adopt this emancipated position with respect to wrapping operations, and formulate the logic of discontinuity in terms of multimodal interaction principles. Consider the Dutch Verb Raising construction. In Ex 2.22 we saw that a *unimodal* 'Mixed Composition' law causes permutation disturbances in an otherwise order-sensitive grammar logic. With the aid of the MP/MA interaction principles, one obtains the multimodal version of Ex 4.4.

Example 4.4 *Mixed Composition/Geach as a multimodal theorem ([Moortgat & Oehrle 94]). The MP interaction principle relates the head adjunction mode \bullet_h , which provides typing for the verb-raising triggers, and the dependency mode \bullet_r , which characterizes the head-final basic clausal structure of Dutch. (Compare $(vp/hiv, np\backslash_riv)^h \Rightarrow np\backslash_rvp$ with Ex 2.22.)*

$$\frac{\frac{\frac{C \Rightarrow C \quad B \Rightarrow B}{(C, C\backslash_r B)^r \Rightarrow B} \backslash_r L \quad A \Rightarrow A}{(A/hB, (C, C\backslash_r B)^r)^h \Rightarrow A} /_h L}{\frac{(C, (A/hB, C\backslash_r B)^h)^r \Rightarrow A}{(A/hB, C\backslash_r B)^h \Rightarrow C\backslash_r A} \backslash_r R} MP \quad \frac{A/hB \Rightarrow (C\backslash_r A)/_h(C\backslash_r B)}{(A/hB, C\backslash_r B)^h \Rightarrow C\backslash_r A} /_h R$$

Notice that the order sensitivity of the individual modes \bullet_r and \bullet_h is respected: the valid forms of mixed composition form a subset of the composition laws derivable within unimodal **LP**. The principles of Directional Consistency and Directional Inheritance, introduced as theoretical primitives in the rule-based setting of CCG, can be seen here to follow automatically from the individual resource management properties of the modes involved and the distributivity principle governing their communication. Ex 4.4 shows that it is *possible* to derive head adjunction. In order to *force* the formation of the verb cluster, the type language has to be further refined. See [Moortgat & Oehrle 94] for discussion, and §4.2 for the required logical vocabulary.

For a second illustration, we take up the discussion of *in situ* binding of §3. It is shown in [Morrill 94a] that the connective $q(A, B, C)$ can be *defined* in a multimodal system with three communicating modes: a (associative regime), n (non-associative regime), and w (wrapping). The crucial interaction principle is given in (22). The deconstruction of Ex 4.5 partially answers the question raised in §3: for a default *associative* regime, it shows how one can define an *in situ* binding operator as $(s/wnp)\backslash_w s$. Associativity here is essential for obtaining access to arbitrary infixation points for the wrapping expression.

$$(WN) : (A \bullet_a B) \bullet_a C \longleftrightarrow (A \bullet_n C) \bullet_w B \quad \frac{\Gamma[((\Delta_1, \Delta_3)^n, \Delta_2)^w] \Rightarrow A}{\Gamma[((\Delta_1, \Delta_2)^a, \Delta_3)^a] \Rightarrow A} [WN] \quad (22)$$

Example 4.5 *Multimodal deconstruction of $q(A, B, C)$ as $(B/wA) \setminus_w C$. On the left the $[qL]$ rule of Def 3.14. On the right the ‘partial execution’ compilation in terms of interaction principle (22).*

$$\frac{\Delta[x : A] \Rightarrow t : B \quad \Gamma[y : C] \Rightarrow u : D}{\Gamma[\Delta[z : q(A, B, C)]] \Rightarrow u[z(\lambda x.t)/y] : D} (qL) \quad \frac{\frac{\frac{(\Delta, (A, \Delta')^a)^a \Rightarrow B}{((\Delta, \Delta')^n, A)^w \Rightarrow B} WN}{(\Delta, \Delta')^n \Rightarrow B/wA} /_w R \quad \Gamma[C] \Rightarrow D}{\Gamma[((\Delta, \Delta')^n, (B/wA) \setminus_w C)^w] \Rightarrow D} \setminus_w L}{\frac{\Gamma[((\Delta, (B/wA) \setminus_w C)^a, \Delta')^a] \Rightarrow D}{\Gamma[(\Delta, q(A, B, C))^a, \Delta')^a] \Rightarrow D} WN} (def)$$

INTERACTION PRINCIPLES: STRONG DISTRIBUTIVITY. As remarked above, the weak distributivity principles MP, MA keep us within the family of resource neutral logics: they do not affect the multiplicity of the resources in a configuration. Strong distributivity principles are not resource neutral: they duplicate resources. As an example, consider the interaction principle of Mixed Contraction in Def 4.6, which strongly distributes mode j over mode i thus copying a C datum. Rather than introducing global Contraction, this interaction principle allows for a constrained form of copying, restricted to the case where modes i and j are in construction.

Definition 4.6 *Restricted Contraction. Structural postulate, Gentzen rule, frame constraint.*

$$MC : (A \bullet_i B) \bullet_j C \rightarrow (A \bullet_j C) \bullet_i (B \bullet_j C) \quad \frac{\Gamma[((\Delta_1, \Delta_3)^j, (\Delta_2, \Delta_3)^j)^i] \Rightarrow A}{\Gamma[(\Delta_1, \Delta_2)^i, \Delta_3)^j] \Rightarrow A} MC$$

$$(R_i t x y \ \& \ R_j u t z) \Rightarrow \exists t' \exists t'' (R_j t' x z \ \& \ R_j t'' y z \ \& \ R_i u t' t'')$$

It has been argued that grammatical inference requires restricted access to Contraction for the analysis of the so-called parasitic gap constructions in (23) below. In this construction, one would like the abstractor associated with the *wh* element to bind multiple occurrences of the same variable, for the interpretation of the structural positions indicated by the underscores. Such multiple binding is beyond the scope of occurrence-sensitive logics we have considered so far. In the framework of CCG, parasitic gaps are handled by means of the combinator **S** which is introduced as a primitive for this purpose, cf. [Szabolcsi 87, Steedman 87].

$$\mathbf{S} : A/C, (A \setminus B)/C \Rightarrow B/C \quad \text{Which books did John (file _ without reading _)} \quad (23)$$

In a multimodal framework, a mode-restricted form of the **S** combinator can be derived from the strong distributivity principle discussed above. In the Gentzen proof below, we give the relevant instance for the derivation of the example sentence (instantiate A/jC as vp/jnp for *file*, and $(A \setminus_i B)/jC$ as $(vp \setminus_i vp)/jnp$ for *without reading*). Mode j here would be the default mode by which the transitive verbs *file* and *read* consume their direct objects; the combination of the *vp* adjunct *without reading* with the *vp* it modifies is given in terms of mode i , the ‘parasitic’ mode which licenses the secondary gap depending on the primary one, the argument of *file*.

Example 4.7 *Deriving the combinator **S** as a multimodal theorem.*

$$\frac{\frac{\&c}{((A/jC, C)^j, (A \setminus_i B)/jC, C)^j \Rightarrow B} MC}{\frac{((A/jC, (A \setminus_i B)/jC)^i, C)^j \Rightarrow B}{(A/jC, (A \setminus_i B)/jC)^i \Rightarrow B/jC} /_j R}$$

4.2 Grammatical composition: unary operations

The language of binary multiplicative connectives is designed to talk about forms of linguistic composition where two resources are put together. It is not difficult to see how one could generalize the type language to n -ary multiplicatives, and interpret families of n -ary residuated connectives with respect to a composition relation of arity $n+1$, in the setting of frame semantics. Writing $f_\bullet(A_1, \dots, A_n)$ for an n -ary product and $f_{\rightarrow}^i(A_1, \dots, A_n)$ for the i -th place residual, the basic residuation laws take the form shown in (24). For arities $2 \leq n$, an n -ary product connective would be interpreted with respect to a form of grammatical composition relating n ‘component’ resources to their ‘fusion’. Such generalizations have been studied in a logical setting in [Dunn 93], and in the context of categorial grammar logics in [Buszkowski 84, Moortgat & Oehrle 93].

$$f_\bullet(A_1, \dots, A_n) \rightarrow B \quad \text{iff} \quad A_i \rightarrow f_{\rightarrow}^i(A_1, \dots, A_{i-1}, B, A_{i+1}, \dots, A_n) \quad (24)$$

In this section we present the logic of *unary* residuated operations in the categorial type language. The need for unary complementation of the familiar binary vocabulary has long been felt: for arguments see [Bach 88], or [Schmerling 83], who relates the discussion to the ‘item-and-arrangement’ versus ‘item-and-process’ views on structuring linguistic resources. As remarked above, unary connectives were introduced in the type-logical discussion around 1990 in [Morrill 90a], and subsequent work of a number of Edinburgh researchers. A representative collection of papers can be found in [Barry & Morrill 90].

Our aim in this section is to systematize this area of research by developing a general framework that will naturally accommodate the various proposals for unary operators while at the same time providing more fine-grained notions of resource control. We extend the language of binary multiplicatives with a pair of unary residual operators \diamond , \square . Parallel to our treatment of the binary multiplicatives §2, we start from the most discriminating system, i.e. the pure logic of residuation for \diamond , \square . By gradually adding structural postulates, we obtain versions of these unary operators with a coarser resource management regime. We develop the model-theoretic and proof-theoretic technicalities in §4.2.1, drawing heavily on [Moortgat 95]. In §4.2.2, we discuss the linguistic motivation for the various resource management options. Finally, in §4.2.3, we present a general theory of structural control in terms of embedding theorems connecting resource management regimes.

4.2.1 Unary connectives: logic and structure

Consider first the pure logic of residuation for a pair of unary type-forming operators \diamond , \square .

Definition 4.8 *Unary multiplicative connectives: the pure logic of residuation. Interpretation clauses. Residuation laws. Note that the interpretation of \diamond and \square ‘moves’ in opposite directions along the R^2 accessibility relation. (The downarrow on the universal operator is there to highlight this fact.)*

$$\begin{aligned} v(\diamond A) &= \{x \mid \exists y (Rxy \wedge y \in v(A))\} \\ v(\square A) &= \{x \mid \forall y (Ryx \Rightarrow y \in v(A))\} \\ \diamond A \rightarrow B &\quad \text{iff} \quad A \rightarrow \square B \end{aligned}$$

COMPLETENESS. The completeness result of Prop 2.3 for the binary multiplicative language extends unproblematically to the language enriched with \diamond , \square . We interpret now with respect to mixed frames, where a binary and a ternary composition relation live together, and consider models $\mathcal{M} = \langle W, R^2, R^3, v \rangle$. In the formula-based canonical model construction of Def 2.4, one defines $R^2(A, B)$ iff $\vdash A \rightarrow \diamond B$. The Truth Lemma has to be checked for the new compound formulae $\diamond A$, $\square A$. The direction that requires a little thinking is dealt with below.

(\diamond) Assume $A \in v(\diamond B)$. We have to show $\vdash A \rightarrow \diamond B$. $A \in v(\diamond B)$ implies $\exists A'$ such that $R^2 AA'$ and $A' \in v(B)$. By induction hypothesis, $\vdash A' \rightarrow B$. By Isotonicity for \diamond (cf. (26) below) this implies $\vdash \diamond A' \rightarrow \diamond B$. We have $\vdash A \rightarrow \diamond A'$ by (Def R^2) in the canonical frame. By Transitivity, $\vdash A \rightarrow \diamond B$.

(\square) Assume $A \in v(\square B)$. We have to show $\vdash A \rightarrow \square B$. $A \in v(\square B)$ implies that $\forall A'$ such that $R^2 A' A$ we have $A' \in v(B)$. Let A' be $\diamond A$. $R^2 A' A$ holds in the canonical frame since $\vdash \diamond A \rightarrow \diamond A$. By induction hypothesis we have $\vdash A' \rightarrow B$, i.e. $\vdash \diamond A \rightarrow B$. By Residuation this gives $\vdash A \rightarrow \square B$.

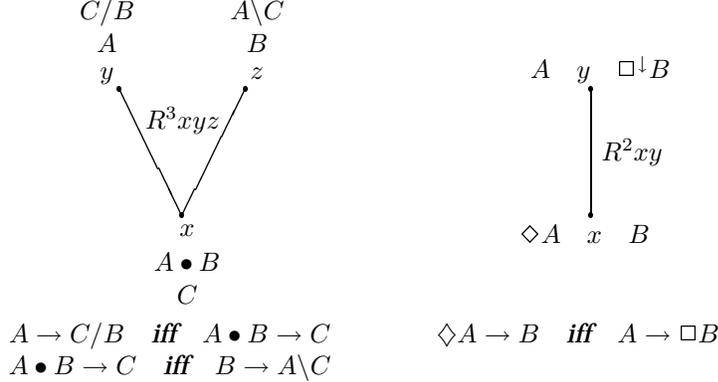


Figure 1: Kripke graphs: binary and unary multiplicatives

Figure 1 may clarify the relation between the unary and the binary residuated pairs of connectives. Notice that if one were interpreting R^2 as temporal priority, \diamond and \square would be interpreted as past possibility and future necessity, respectively. But in the grammatical application, R^2 just like R^3 is to be interpreted in terms of structural composition. Where a ternary configuration $(xyz) \in R^3$ abstractly represents putting together the components y and z into a structured configuration x in the manner indicated by R^3 , a binary configuration $(xy) \in R^2$ can be seen as the construction of the sign x out of a single structural component y in terms of the building instructions referred to by R^2 . (An ‘additive’ alternative to the ‘multiplicative’ view on unary operators presented here, will be presented in Def ??).

In our discussion of the binary vocabulary in §2, we pointed out that one can characterize $/, \bullet, \setminus$ as a residuated family either in terms of the basic law RES of Def 2.2, or in terms of the (Co-)Application and Monotonicity laws of Prop 2.18. Similarly, for the unary connectives, we have the equivalent Lambek-style and Došen-style axiomatizations of Def 4.9.

Definition 4.9 *Unary connectives: alternative combinator presentations.* (†) Lambek-style in terms of Residuation. (‡) Došen-style in terms of compositions $\diamond \square$, $\square \diamond$, and Isotonicity.

$$\begin{array}{l}
(\dagger) \quad \frac{f : \diamond A \rightarrow B}{\mu(f) : A \rightarrow \square B} \quad \frac{g : A \rightarrow \square B}{\mu^{-1}(g) : \diamond A \rightarrow B} \\
(\ddagger) \quad \begin{array}{ll}
\mathbf{unit}_{\square} : \diamond \square A \rightarrow A & \frac{f : A \rightarrow B}{(f)^{\diamond} : \diamond A \rightarrow \diamond B} \\
\mathbf{co-unit}_{\square} : A \rightarrow \square \diamond A & \frac{f : A \rightarrow B}{(f)^{\square} : \square A \rightarrow \square B}
\end{array}
\end{array}$$

We take the Lambek-style presentation as our starting point here, and show for the extended system how from the residuation inferences μ, μ^{-1} we obtain the alternative axiomatization in terms of Isotonicity and the inequalities for the compositions $\diamond \square$ and $\square \diamond$ (Term decoration for the right column left to the reader.)

$$\frac{1_{\square A} : \square A \rightarrow \square A}{\mu^{-1}(1_{\square A}) : \diamond \square A \rightarrow A} \quad \frac{\diamond A \rightarrow \diamond A}{A \rightarrow \square \diamond A} \quad (25)$$

$$\frac{f : A \rightarrow B \quad \frac{1_{\diamond B} : \diamond B \rightarrow \diamond B}{\mu(1_{\diamond B}) : B \rightarrow \square \diamond B}}{\mu(1_{\diamond B}) \circ f : A \rightarrow \square \diamond B} \quad \frac{\frac{\square A \rightarrow \square A}{\diamond \square A \rightarrow A} \quad A \rightarrow B}{\frac{\diamond \square A \rightarrow B}{\square A \rightarrow \square B}} \quad (26)$$

GENTZEN CALCULUS. Following the agenda set out in §2 for the binary connectives, we introduce Gentzen sequent rules for the connectives \diamond, \square . Corresponding to the formula language \mathcal{F} of (27) we have a language of Gentzen terms \mathcal{S} for structured configurations of formulae. Gentzenization for the extended type language requires an n -ary structural operator for every family of n -ary logical operators: binary (\cdot, \cdot) for the family $/, \bullet, \setminus$, and unary $(\cdot)^\diamond$ for the family \diamond, \square .

$$\mathcal{F} ::= \mathcal{A} \mid \mathcal{F}/\mathcal{F} \mid \mathcal{F} \bullet \mathcal{F} \mid \mathcal{F} \setminus \mathcal{F} \mid \diamond \mathcal{F} \mid \square \mathcal{F} \quad \mathcal{S} ::= \mathcal{F} \mid (\mathcal{S}, \mathcal{S}) \mid (\mathcal{S})^\diamond \quad (27)$$

Definition 4.10 *Unary connectives: Gentzen rules. Belnap-style antecedent punctuation, with unary structural connective $(\cdot)^\diamond$ matching the unary logical connective \diamond .*

$$\frac{\Gamma \Rightarrow A}{(\Gamma)^\diamond \Rightarrow \diamond A} \diamond R \quad \frac{\Gamma[(A)^\diamond] \Rightarrow B}{\Gamma[\diamond A] \Rightarrow B} \diamond L$$

$$\frac{(\Gamma)^\diamond \Rightarrow A}{\Gamma \Rightarrow \square A} \square R \quad \frac{\Gamma[A] \Rightarrow B}{\Gamma[(\square A)^\diamond] \Rightarrow B} \square L$$

As shown in [Moortgat 95], the Gentzen presentation is equivalent to the axiomatization of Def 4.9, and it allows Cut Elimination with its pleasant corollaries: decidability and the subformula property.

UNARY CONNECTIVES: STRUCTURAL POSTULATES. Completeness for the pure logic of residuation for the unary family \diamond, \square does not depend on semantic restrictions on the R^2 composition relation. In addition to the fixed ‘logical’ part of the \diamond, \square connectives, we can consider various structural resource management options for the unary family \diamond, \square and its binary accessibility relation R^2 , and for the mixed R^2, R^3 system.

The structural postulates in Def 4.11 constrain R^2 to be transitive (4), or reflexive (T). Communication between R^2 and R^3 can be established via the ‘percolation’ principles $K(1, 2)$. The strong distributivity postulate K distributes unary \diamond over both components of a binary \bullet . The more constrained weak distributivity postulates $K1, K2$ make \diamond select the left or right subtype of a product. The combination of the options $KT4$ gives an $S4$ modality with the logical rules of use and proof of the Linear Logic exponential ‘!’.

Observe that the postulates have the required Weak Sahlqvist form for the extended completeness result of Prop 2.7. In [Moortgat 95], the Cut Elimination result for the pure residuation logic of Def 4.10 is extended to cover the structural options of Def 4.11. In a multimodal setting, one can further enhance the linguistic expressivity by combining different composition modes R_j^2 for $\langle j \rangle, [j]^1$ in one logic. The multimodal generalization is completely standard.

Definition 4.11 *Unary connectives: resource management options. Structural postulates, frame constraints, Gentzen rules. (For \square duals of these postulates: replace \diamond by \square and reverse the arrow.)*

$$\begin{array}{ll} 4 : & \diamond \diamond A \rightarrow \diamond A \quad (Rxy \ \& \ Ryz) \Rightarrow Rxz \\ T : & A \rightarrow \diamond A \quad Rxx \\ K1 : & \diamond(A \bullet B) \rightarrow \diamond A \bullet B \quad (Rwx \ \& \ Rxyz) \Rightarrow \exists y' (Ry'y \ \& \ Rwy'z) \\ K2 : & \diamond(A \bullet B) \rightarrow A \bullet \diamond B \quad (Rwx \ \& \ Rxyz) \Rightarrow \exists z' (Rz'z \ \& \ Rwyz') \\ K : & \diamond(A \bullet B) \rightarrow \diamond A \bullet \diamond B \quad (Rwx \ \& \ Rxyz) \Rightarrow \exists y' \exists z' (Ry'y \ \& \ Rz'z \ \& \ Rwy'z') \end{array}$$

$$\frac{\Gamma[(\Delta)^\diamond] \Rightarrow A}{\Gamma[(\Delta)^\diamond] \Rightarrow A} 4 \quad \frac{\Gamma[(\Delta)^\diamond] \Rightarrow A}{\Gamma[\Delta] \Rightarrow A} T$$

$$\frac{\Gamma[(\Delta_1)^\circ, \Delta_2] \Rightarrow A}{\Gamma[(\Delta_1, \Delta_2)]^\circ \Rightarrow A} K1 \quad \frac{\Gamma[(\Delta_1)^\circ, (\Delta_2)^\circ] \Rightarrow A}{\Gamma[(\Delta_1, \Delta_2)]^\circ \Rightarrow A} K \quad \frac{\Gamma[(\Delta_1, (\Delta_2)^\circ)] \Rightarrow A}{\Gamma[(\Delta_1, \Delta_2)]^\circ \Rightarrow A} K2$$

S4: COMPILATION OF STRUCTURAL RULES. We saw in Def 2.13 that in the presence of Associativity for \bullet , we have a sugared Gentzen presentation where the structural rule is compiled away, and the binary sequent punctuation (\cdot, \cdot) omitted. Analogously, for \square with the combination $KT4$ (i.e. $S4$), we have a sugared version of the Gentzen rules, where the $KT4$ structural rules are compiled away, so that the unary $(\cdot)^\circ$ punctuation can be omitted. In the sugared version, we recognize the rules of use and proof for the domain modalities of [Morrill 90a, Hepple 90].

Definition 4.12 *Sugared presentation of $KT4$ modalities: compiling out the $(\cdot)^\circ$ structural punctuation.* We write $\square\Gamma$, $(\square)^\circ\Gamma$, $(\square\square)^\circ\Gamma$ for a term Γ of which the (pre)terminal subterms are all of the form $\square A$, $(\square A)^\circ$, $(\square\square A)^\circ$, respectively. The $4(Cut)$ step is a series of replacements (read bottom-up) of terminal $\square A$ by $\square\square A$ via Cuts depending on 4.

$$\frac{\frac{\Gamma[A] \Rightarrow B}{\Gamma[(\square A)^\circ] \Rightarrow B} \square L}{\Gamma[\square A] \Rightarrow B} T \quad \rightsquigarrow \quad \frac{\Gamma[A] \Rightarrow B}{\Gamma[\square A] \Rightarrow B} \square L(S4)$$

$$\frac{\frac{\frac{\square\Gamma \Rightarrow A}{(\square\square)^\circ\Gamma \Rightarrow A} \square L}{(\square)^\circ\Gamma \Rightarrow A} 4(Cut)}{\frac{(\square\Gamma)^\circ \Rightarrow A}{\square\Gamma \Rightarrow \square A} K} \square R \quad \rightsquigarrow \quad \frac{\square\Gamma \Rightarrow A}{\square\Gamma \Rightarrow \square A} \square R(S4)$$

SITUATING UNARY OPERATORS. The above analysis of the unary vocabulary in its logical and structural components provides us with a tool to evaluate existing proposals for unary operators. In doing so, we follow the methodological ‘minimality’ principle adopted above in the discussion of the binary vocabulary, i.e. we try to pinpoint exactly which assumptions about the composition relation are necessary to achieve a certain grammatical effect.

At one end of the spectrum, the proposals that come closest to the pure logic of residuation for \diamond, \square are the ‘bracket’ operators of [Morrill 95a, Morrill 94a]. On the semantic level, the bracket operators are given an algebraic interpretation which, in the context of frame semantics, would amount to a functionality requirement for the accessibility relation R^2 . The linguistic applications of the bracket operators as markers of locality domains can be recast straightforwardly in terms of the more discriminating pure residuation logic for \diamond, \square for which a sound and complete logic is available, imposing no functionality constraints on R^2 .

At the other end of the spectrum, we find the domain modalities of [Morrill 90a, Hepple 90], universal \square operators which assume the full set of postulates $KT4$ i.e. $S4$. Adding modally controlled structural rules, we obtain the structural modalities of [Barry e.a. 91, Morrill 94a]. Like the exponentials of Linear Logic, the structural modalities license controlled access to resource management options that are not freely available. As we will see in §??, the $S4$ logical rules are *incomplete* with respect to the intended subalgebra semantics for these connectives. Again, we can scrutinize the $S4$ assumptions, and see whether a more delicate resource management regime can achieve the same effects.

In the framework presented here, where we consider a residuated *pair* of modalities \diamond, \square rather than a single modal operator \square , we can *simulate* the T and 4 postulates proof-theoretically, without making Reflexivity or Transitivity assumptions about the R^2 composition relation. With the translation of Def 4.13 the images of the T and 4 postulates for \square become valid type transitions in the pure residuation system for \diamond, \square , as the reader can check. For modally controlled structural rules, Def 4.14 gives restricted versions of the global rules keyed to \diamond contexts; for communication between the unary and binary multiplicatives, one can rely on the K distributivity principles.

Definition 4.13 *Simulating T and Δ via compilation* $(\Box A)^\# = \Diamond \Box (A)^\#$.

$$\begin{aligned} T : \Box A \rightarrow A &\quad \rightsquigarrow \quad \Diamond \Box A \rightarrow A \\ \Delta : \Box A \rightarrow \Box \Box A &\quad \rightsquigarrow \quad \Diamond \Box A \rightarrow \Diamond \Box \Box A \end{aligned}$$

Definition 4.14 *Modally restricted structural options: Commutativity (P_\diamond), Associativity (A_\diamond). Structural postulates, Gentzen rules. The side condition (\dagger) requires one of the A_i (Δ_i) to be of the form $\Diamond A$ ($(\Delta)^\diamond$).*

$$(P_\diamond) : \Diamond A \bullet B \rightarrow B \bullet \Diamond A \quad (A_\diamond) : (A_1 \bullet A_2) \bullet A_3 \longleftrightarrow A_1 \bullet (A_2 \bullet A_3)(\dagger)$$

$$\frac{\Gamma[(\Delta_2)^\diamond, \Delta_1] \Rightarrow A}{\Gamma[(\Delta_1, (\Delta_2)^\diamond)] \Rightarrow A} (P_\diamond) \quad (\dagger) \frac{\Gamma[(\Delta_1, \Delta_2), \Delta_3] \Rightarrow A}{\Gamma[(\Delta_1, (\Delta_2, \Delta_3))] \Rightarrow A} (A_\diamond)$$

TERM ASSIGNMENT: UNARY CONNECTIVES. To close this section, we present the term assignment for the unary connectives in an abstract format, with constructor/destructor operations in the term language matching rules of use and proof.

Definition 4.15 *Syntax of typed lambda terms: clauses for \Diamond, \Box . Destructors ${}^\cup$ and ${}^\vee$, corresponding to rules of use for \Diamond and \Box . Constructors ${}^\cap$ and ${}^\wedge$, for rules of proof. Compare Def 3.1 for the binary vocabulary.*

$$\mathcal{M}^A ::= \dots \mid {}^\cup(\mathcal{M}^{\Diamond A}) \mid {}^\vee(\mathcal{M}^{\Box A}) \quad \mathcal{M}^{\Diamond A} ::= {}^\cap(\mathcal{M}^A) \quad \mathcal{M}^{\Box A} ::= {}^\wedge(\mathcal{M}^A)$$

Definition 4.16 *Term assignment. The \Diamond, \Box cases.*

$$\begin{aligned} \frac{\Gamma \Rightarrow t : A}{(\Gamma)^\diamond \Rightarrow {}^\cap t : \Diamond A} \Diamond R &\quad \frac{\Gamma[(y : A)^\diamond] \Rightarrow t : B}{\Gamma[x : \Diamond A] \Rightarrow t[{}^\cup x/y] : B} \Diamond L \\ \frac{(\Gamma)^\diamond \Rightarrow t : A}{\Gamma \Rightarrow {}^\wedge t : \Box A} \Box R &\quad \frac{\Gamma[y : A] \Rightarrow t : B}{\Gamma[(x : \Box A)^\diamond] \Rightarrow t[{}^\vee x/y] : B} \Box L \end{aligned}$$

Definition 4.17 *Term equations and their Gentzen proof-theoretic reflexes. Compare the binary case in Def 3.5.*

$$\begin{aligned} {}^\cup({}^\cap t) = t &\quad \rightsquigarrow \quad \text{principal cut on } \Diamond A & \quad {}^\vee({}^\wedge t) = t &\quad \rightsquigarrow \quad \text{principal cut on } \Box A \\ {}^\cap({}^\cup t) = t &\quad \rightsquigarrow \quad \text{non-atomic axiom } \Diamond A & \quad {}^\wedge({}^\vee t) = t &\quad \rightsquigarrow \quad \text{non-atomic axiom } \Box A \end{aligned}$$

Concrete realizations of the abstract term assignment schema will depend on the application. For an example, we refer to the type-logical implementation of Montague-style intensional semantics driven from an $S4$ universal modality in [Morrill 90a]. Let us write the ‘intensionality’ type-forming operator as \Box . We interpret formulas $\Box A$ as functions from indices to the denotata of formulas A . Term assignment for the rules of use and proof for \Box can then be given in terms of Montague’s ‘cup’ and ‘cap’ operations, respectively. Cf. Chapter One.

$$\frac{\Box \Gamma \Rightarrow t : A}{\Box \Gamma \Rightarrow {}^\wedge t : \Box A} \Box R \quad \frac{\Gamma, x : A, \Gamma' \Rightarrow t : B}{\Gamma, y : \Box A, \Gamma' \Rightarrow t[{}^\vee y/x] : B} \Box L \quad (28)$$

For another application, we refer to the work on information packaging in [Hendriks 94], where the term assignment for \Diamond realizes the prosodic and pragmatic structuring of the text in terms of stress and given/new distinctions.

4.2.2 Applications: imposing constraints, structural relaxation

One can develop two perspectives on controlling resource management, depending on the direction of communication. On the one hand, one would like to have control devices to license limited access to a more liberal resource management regime from within a system with a higher sense of structural discrimination. On the other hand, one would like to impose constraints on resource management in systems where such constraints are lacking by default.

LICENSING STRUCTURAL RELAXATION. For the licensing type of communication, consider type assignment $r/(s/np)$ to relative pronouns like *that* in the sentences below.

$$\begin{array}{l}
 \text{(the book) that Kazimierz wrote} \\
 \text{(the book) that Kazimierz wrote yesterday} \\
 \mathbf{L} \vdash r/(s/np), np, (np \setminus s)/np \Rightarrow r \\
 \mathbf{L} \not\vdash r/(s/np), np, (np \setminus s)/np, s \setminus s \Rightarrow r \\
 \mathbf{NL} \not\vdash (r/(s/np), (np, (np \setminus s)/np)) \Rightarrow r
 \end{array} \tag{29}$$

Suppose first we are dealing with the associative regime of **L**. The first example is derivable, the second is not because the hypothetical np assumption in the subderivation ‘Kazimierz wrote yesterday np ’ is not in the required position adjacent to the verb ‘wrote’. We can refine the assignment to the relative pronoun to $r/(s/!_c np)$, where $!_c np$ is a noun phrase resource which has access to Permutation in virtue of its modal decoration. Similarly, if we change the default regime to **NL**, the first example already fails on the assignment $r/(s/np)$ with the indicated constituent bracketing: although the hypothetical np in the subcomputation ‘((Kazimierz wrote) np)’ finds itself in the right position with respect to linear order requirements, it cannot satisfy the direct object role for ‘wrote’ being outside the clausal boundaries. A refined assignment $r/(s/!_a np)$ here could license the marked $!_a np$ a controlled access to the structural rule of Associativity which is absent in the **NL** default regime.

As remarked above, cases like these have been handled in terms of $S4$ -style structural modalities in [Barry & Morrill 90, Morrill 94a]. In (30), we illustrate the deconstruction of $!$ as $\diamond \square$ with the derivation of controlled rebracketing within **NL**.

$$\begin{array}{c}
 \frac{}{\&c} \\
 \frac{}{(np, (tv, np)) \Rightarrow s} \\
 \frac{(np, (tv, (\square_a np)^\diamond)) \Rightarrow s}{(np, (tv, (\square_a np)^\diamond)) \Rightarrow s} \square L \\
 \frac{(np, tv), (\square_a np)^\diamond \Rightarrow s}{((np, tv), (\square_a np)^\diamond) \Rightarrow s} A_\diamond \\
 \frac{((np, tv), \diamond_a \square_a np) \Rightarrow s}{(np, tv), \diamond_a \square_a np \Rightarrow s} \diamond L \\
 \frac{(np, tv) \Rightarrow s / \diamond_a \square_a np}{(np, tv) \Rightarrow s / \diamond_a \square_a np} /R
 \end{array} \tag{30}$$

IMPOSING STRUCTURAL CONSTRAINTS. For the other direction of communication, we return to the violations of the Coordinate Structure Constraint, discussed in §2 in connection with the over-generation of **L**. Consider the relative clauses of Ex 4.18. With the instantiation $X = s/np$ for the polymorphic conjunction particle, we can derive the (a) example. But, given Associativity and an instantiation $X = s$, nothing blocks the derivation of the ungrammatical (b) example.

Example 4.18 *Lexical projection of island constraints ([Morrill 95a, Morrill 94a]).*

- a. *(the logician) whom Gottlob admired and Kazimierz detested*
 $\mathbf{L} \vdash r/(s/np), np, tv, (X \setminus X)/X, np, tv \Rightarrow r \quad (X = s/np)$
 $\mathbf{L} \diamond \vdash r/(s/np), (np, tv, (X \setminus \square X)/X, np, tv)^\diamond \Rightarrow r$
- b. **(the logician) whom Gottlob admired Jim and Kazimierz detested*
 $\mathbf{L} \vdash r/(s/np), np, tv, np, (X \setminus X)/X, np, tv \Rightarrow r \quad (X = s)$
 $\mathbf{L} \diamond \not\vdash r/(s/np), (np, tv, np, (X \setminus \square X)/X, np, tv)^\diamond \Rightarrow r$

In [Morrill 95a, Morrill 94a] it is shown that the coordinate structure domain can be lexically projected from a modal refinement of the assignment to ‘and’: $(X \setminus \Box X) / X$. (We recast the analysis in terms of the pure residuation logic for \diamond, \Box .) The refined assignment allows the conjunction to combine with the left and right conjuncts in the associative mode. The resulting coordinate structure is of type $\Box X$. To eliminate the \Box connective, we have to close off the coordinate structure with \diamond (or the corresponding structural operator $(\cdot)^\diamond$ in the Gentzen presentation) — recall the basic reduction $\diamond \Box X \rightarrow X$. The Accross-the-Board case of extraction (4.18a) works out fine, the island violation (4.18b) fails because the hypothetical gap np assumption finds itself outside the scope of the $(\cdot)^\diamond$ operator.

In [Versmissen 96], this use of modal decoration is generalized into a type-logical formulation of the theory of word-order domains of [Reape 89]. The control operators \diamond, \Box provide a fully general vocabulary for projection and erasure of domains of locality, according to the following scheme distinguishing the antecedent (resource) versus succedent (goal) effects of \diamond, \Box decoration.

	RESOURCE	GOAL
\diamond	domain-erasure	domain-projection
\Box	domain-projection	domain-erasure

(31)

MODALITIES AS DOMAINS OF LOCALITY. In [Morrill 90a], locality domains, in the sense of semantic intensionality, are characterized in terms of a uniform $S4 \Box$ decoration for the resources that make up a domain, cf. (28). [Hepple 90], dropping the semantic component of this proposal, uses the \Box decoration to capture syntactic boundary effects. These applications are instructive because they crucially rely on the rule of *proof* for the $S4$ universal modality: as we have seen in Def 4.12, this rule insists that all assumptions on which a $\Box A$ formula depends are themselves \Box decorated.

Consider the constraint of clause-boundedness that governs the use of the English reflexive pronouns. In Ex 3.12 we discussed an **L** type-assignment $((np \setminus s) / np) \setminus (np \setminus s)$ for ‘himself’ with meaning recipe $\lambda x \lambda y. x(y)(y)$. Within **L**, (a), (b) and (c) are all derivable: this system, because of the global availability of associativity, cannot discriminate between a lexical or complex clause-internal expression of type $((np \setminus s) / np)$ and a complex expression of that type which has been composed across clausal boundaries.

- | | | | |
|--|----------|---|------|
| a. David <u>admires</u> himself | L | ⊢ $(np \setminus s) / np \Rightarrow (np \setminus s) / np$ | |
| b. David <u>cares</u> for himself | L | ⊢ $(np \setminus s) / pp, pp / np \Rightarrow (np \setminus s) / np$ | (32) |
| c. *David <u>thinks Emmy admires</u> himself | L | ⊢ $(np \setminus s) / s, np, (np \setminus s) / np \Rightarrow (np \setminus s) / np$ | |

Within **L**+ \Box , appropriate modalization provides lexical control to make (a) and (b) derivable while ruling out (c). In moving from **L** to **L**+ \Box lexical type assignments, one prefixes the original **L** lexical assignments with a \Box operator, and further decorates with a \Box every argument subtype B that constitutes a locality domain. The effect of such modalization for the lexical resources of (32c) is shown in Ex 4.19.

Example 4.19 *Blocking locality violations via $S4 \Box$ decoration ([Morrill 90a, Hepple 90]). The assignment to the verb ‘think’ marks its clausal complement as a locality domain. The derivation for the non-local reading (32c) fails, because the hypothetical direct object np assumption is not decorated with \Box , blocking application of the $[\Box R]$ inference, which requires all the antecedent assumptions on which it depends to be modally marked.*

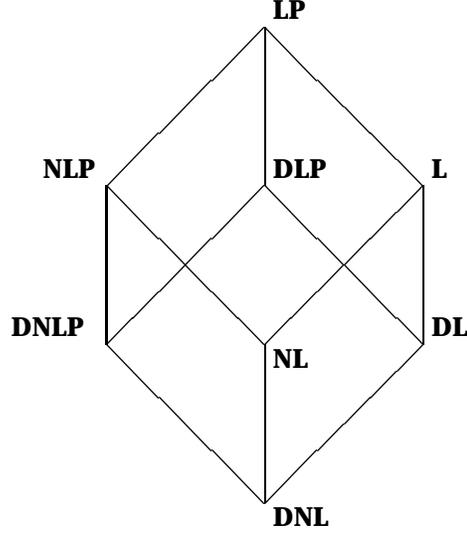


Figure 2: Resource-sensitive logics: precedence, dominance, dependency

a structural rule that would be applicable otherwise. For the \cdot^\sharp direction, the modal decoration gives access to a controlled version of a structural rule which is unavailable in its ‘global’ (non-decorated) version.

We illustrate the two-way structural control with the pair **NL** and **L**. Let us subscript the connectives in **NL** with 0 and those of **L** with 1. The embedding translations \cdot^b and \cdot^\sharp are given in Def 4.20. For the two directions of communication, the same decoration schema can be used.

Definition 4.20 *Embedding translations* $\cdot^b : \mathcal{F}(\mathbf{NL}) \mapsto \mathcal{F}(\mathbf{L}\diamond)$, and $\cdot^\sharp : \mathcal{F}(\mathbf{L}) \mapsto \mathcal{F}(\mathbf{NL}\diamond)$.

$$\begin{array}{ll}
 p^b = p & p^\sharp = p \\
 (A \bullet_0 B)^b = \diamond(A^b \bullet_1 B^b) & (A \bullet_1 B)^\sharp = \diamond(A^\sharp \bullet_0 B^\sharp) \\
 (A/_0 B)^b = \square A^b/_1 B^b & (A/_1 B)^\sharp = \square A^\sharp/_0 B^\sharp \\
 (B \setminus_0 A)^b = B^b \setminus_1 \square A^b & (B \setminus_1 A)^\sharp = B^\sharp \setminus_0 \square A^\sharp
 \end{array}$$

The **L** system has an associative resource management which is insensitive to constituent bracketing. Extending **L** with the operators \diamond, \square we can recover control over associativity in the sense of Prop 4.21. A conjecture of embedding on the basis of \cdot^b can be found in [Morrill 94b].

Proposition 4.21 *Dominance structure: recovering control* ([Kurtonina & Moortgat 95]).

$$\mathbf{NL} \vdash A \rightarrow B \quad \text{iff} \quad \mathbf{L}\diamond \vdash A^b \rightarrow B^b$$

Consider next the other direction of communication: suppose one wants to obtain the structural flexibility of **L** within the system **NL** with its rigid constituent sensitivity. This time, one achieves the desired embedding result by means of the embedding translation \cdot^\sharp of Def 4.20 together with a modally controlled version of the structural rule of Associativity, relativized to the critical \diamond decoration.

Definition 4.22 *Associativity. Global version* (A) and its image under $(\cdot)^\sharp$, (A_\diamond).

$$\begin{array}{l}
 \mathcal{L}_1 : \quad A \bullet_1 (B \bullet_1 C) \longleftrightarrow (A \bullet_1 B) \bullet_1 C \quad (A) \\
 \mathcal{L}_0 : \quad \diamond(A \bullet_0 \diamond(B \bullet_0 C)) \longleftrightarrow \diamond(\diamond(A \bullet_0 B) \bullet_0 C) \quad (A_\diamond)
 \end{array}$$

5 Reasoning about multiple type assignments

...

6 Hybrid architectures

...

7 Categorical parsing as deduction

...

8 Conclusions, directions for further research

If one compares the present state of the field with the situation as described seven years ago in the overview chapter of [Moortgat 88], one can describe the changes as dramatic. On the level of ‘empirical coverage’, one has witnessed a strong increase in linguistic sophistication. On the level of ‘logical foundations’, Lambek’s architecture for a grammar logic has been significantly generalized without sacrificing the attractive features of the original design. Below we summarize some key themes of the type-logical research that may facilitate comparison with related grammatical frameworks.

- Design of a specific *grammar logic*, i.e. a logic with a consequence relation attuned to the resource-sensitivity of grammatical inference — to be contrasted with ‘general purpose’ specification languages for grammar development, where such resource sensitivity has to be stipulated, e.g. the language of feature logic used in *HPSG*.
- A unified *deductive perspective* on the composition of form and meaning in natural language — to be contrasted with *rule-based* implementations of the compositionality principle.
- Radical lexicalism. Properties of the macro-grammatical organisation are fully projected from lexical type declarations.
- Integration of the grammar logic in a wider landscape of reasoning systems, so that the transition between the formal systems characterizing ‘knowledge of language’ and the systems of inference underlying more general cognitive/reasoning capacities can be seen as gradual.

The change of emphasis from individual type logics to mixed architectures suggests new lines of research. The following themes, among others, would seem relevant for future exploration.

- Descriptive studies. Although, according to [Carpenter 96], the current descriptive coverage of type logical grammar rivals that of competing grammar formalisms, one can expect a wide range of further descriptive studies exploiting the expressivity of interactive modes of structural composition. Contrastive studies could deepen our understanding of the ‘logical’ perspective on parametric variation, characterized in terms of language specific structural rule packages.
- Formal learnability theory. From a cognitive point of view, the radical lexicalism of the type-logical approach makes the *acquisition problem* acute. This requires a theory explaining how multimodal type assignments (with their resource management properties) could be

inferred from exposition to raw data. Learnability theory for the extended type logics can build on results that have been obtained for individual systems (e.g. [Buszkowski & Penn 90, Kanazawa 92], but will have to remove the unrealistic input conditions for the learning algorithm assumed in these studies.

- Computational complexity. Even at the level of individual systems, our knowledge is partial, see [van Benthem 91,95, Aarts 94, Aarts & Trautwein 95] for discussion and results. The context-free recognizing power result for \mathbf{L} of [Pentus 93], for example, has not yielded a polynomial complexity result for this system. In the multimodal setting, one would like to have a systematic theory linking complexity to the algebraic properties of the characterizing rule packages.
- Connections between categorial type-logics and Linear Logic. In this chapter, we have presented an interpretation of the categorial formalism in terms of structural composition of grammatical resources. Recent studies of applications of Linear Logic in linguistic analysis suggest an interesting alternative interpretation in terms of temporal composition of grammatical processes. See [Lecomte & Retoré 95] for an illustration. An integration of these two complementary perspectives would offer a unified framework for the study of the static aspects of linguistic structure and the dynamics of natural language communication.

Acknowledgements

This chapter is based in part on work supported by the National Science Foundation under Grant No. SBR-9510706, and on research conducted in the context of the Esprit BRA project 6852 ‘Dynamic interpretation of natural language’. I thank the editors, Gosse Bouma and Martin Emms for comments on preliminary versions. The gestation period was somewhat longer than anticipated: it started with [Moortgat & Oehrle 93], and produced [Moortgat 95, Kurtonina & Moortgat 95] as preparations for the final delivery. I am deeply indebted to Natasha Kurtonina, Dick Oehrle and Glyn Morrill: it was a pleasure to work together and exchange ideas with them over these years. The chapter has greatly benefited from their efforts. I also wish to thank Herman Hendriks for his meticulous comments which turned final drafts into prefinal ones. All remaining errors and imperfections are my own.

References

- [Aarts 94] Aarts, E. (1994), ‘Proving theorems of the second order Lambek calculus in polynomial time’. *Studia Logica* **53**, 373–387.
- [Aarts & Trautwein 95] Aarts, E. & K. Trautwein (1995), ‘Non-associative Lambek categorial grammar in polynomial time’. *Mathematical Logic Quarterly* **41**, 476–484.
- [Abrusci 96] Abrusci, M.V. (1996), ‘Semantics of proofs for noncommutative linear logic’. Preprint CILA, University of Bari.
- [Abrusci e.a. 94] Abrusci, M., C. Casadio and M. Moortgat (eds.) (1994), *Linear Logic and Lambek Calculus*. Proceedings 1993 Rome Workshop. Esprit BRA 6852 Dyana-2 Occasional Publications, ILLC, Amsterdam.
- [Ades & Steedman 82] Ades, A. & M. Steedman (1982), ‘On the order of words’. *Linguistics & Philosophy* **4**, 517–558.
- [Adriaans 92] Adriaans, P. (1992), *Language Learning from a Categorial Perspective*. Ph.D. Dissertation. University of Amsterdam.

- [Ajdukiewicz 35] Ajdukiewicz, K. (1935), 'Die syntaktische Konnexität.' *Studia Philosophica*, 1:1–27. (English translation in Storrs McCall (ed.) *Polish Logic, 1920–1939*. Oxford (1967), 207–231.)
- [Andréka & Mikulás 94] Andréka, H. & S. Mikulás (1994), 'Lambek calculus and its relational semantics: completeness and incompleteness.' *Journal of Logic, Language, and Information* **3**, 1–38.
- [Andreoli 92] Andreoli, J.-M. (1992) 'Logic programming with focussing proofs in Linear Logic.' *Journal of Logic and Computation* **2**(3).
- [Bach 83a] Bach, E. (1983a), 'On the relationship between word-grammar and phrase-grammar.' *Natural Language & Linguistic Theory* **1**, 65–89.
- [Bach 83b] Bach, E. (1983b), 'Generalized Categorical Grammars and the English Auxiliary'. In F. Heny and B. Richards (eds.) *Linguistic Categories. Auxiliaries and Related Puzzles*. Vol II, Reidel, Dordrecht, 101–120.
- [Bach 84] Bach, E. (1984), 'Some generalizations of categorial grammar'. In F. Landman and F. Veltman (eds.) *Varieties of Formal Semantics*. Foris, Dordrecht, 1–23.
- [Bach 88] Bach, E. (1988), 'Categorial grammars as theories of language'. In [Oehrle e.a. 88], 17–34.
- [Bar-Hillel 64] Bar-Hillel, Y. (1964), *Language and Information*. Addison-Wesley, New York.
- [Barendregt 92] Barendregt, H. (1992), ' λ -Calculi with Types'. In S. Abramsky, D.M. Gabbay, and T.E. Maibaum (eds.) *Handbook of Logic in Computer Science*. Vol 2, 117–309. Oxford.
- [Barry 91] Barry, G. (1991), *Derivation and Structure in Categorical Grammar*. Ph.D. Dissertation, Edinburgh.
- [Barry e.a. 91] Barry, G., M. Hepple, N. Leslie, and G. Morrill (1991), 'Proof figures and structural operators for categorial grammar'. *Proceedings of the Fifth Conference of the European Chapter of the Association for Computational Linguistics*, Berlin.
- [Barry & Morrill 90] Barry, G. and G. Morrill (eds.) (1990), *Studies in Categorical Grammar*. Edinburgh Working Papers in Cognitive Science, Vol 5. CCS, Edinburgh.
- [Barry & Pickering 90] Barry, G. & M. Pickering (1990), 'Dependency and constituency in categorial grammar'. In [Barry & Morrill 90], 23–45.
- [Bayer & Johnson 95] Bayer, S. and M. Johnson (1995), 'Features and agreement' *Proceedings ACL 1995*, San Francisco, 70–76.
- [Belnap 82] Belnap, N.D. (1982), 'Display Logic'. *Journal of Philosophical Logic* **11**, 375–417.
- [van Benthem 83] Benthem, J. van (1983), 'The semantics of variety in categorial grammar'. Report 83–29, Simon Fraser University, Burnaby (B.C.), Canada. (Revised version in [Buszkowski e.a. 88].)
- [van Benthem 84] Benthem, J. van (1984), 'Correspondence theory'. In D. Gabbay and F. Günthner (eds.) *Handbook of Philosophical Logic*. Vol II, Dordrecht, 167–247.
- [van Benthem 87] Benthem, J. van (1987), 'Categorial grammar and lambda calculus'. In D. Skořdev (ed.) *Mathematical Logic and Its Applications*. Plenum, New York, 39–60.
- [van Benthem 88a] Benthem, J. van (1988), 'Categorial grammar meets unification'. In J. Wedekind e.a. (eds.) *Unification Formalisms: Syntax, Semantics and Implementation*.
- [van Benthem 88b] Benthem, J. van (1988), 'The Lambek calculus'. In [Oehrle e.a. 88], 35–68.

- [van Benthem 91,95] Benthem, J. van (1991,1995), *Language in Action. Categories, Lambdas, and Dynamic Logic*. Studies in Logic, North-Holland, Amsterdam. (Student edition: MIT Press (1995), Cambridge, MA.)
- [Blackburn & de Rijke 94] Blackburn, P. and M. de Rijke (1995), 'Zooming-in, zooming-out'. In J. Seligman & D. Westerstahl (eds.) *Logic, Language, and Computation*. CSLI Lecture Notes, Chicago.
- [Bouma 88a] Bouma, G. (1988), *Nonmonotonicity and Categorical Unification Grammar*. Ph.D. Dissertation, Groningen.
- [Bouma 88b] Bouma, G. (1988), 'Modifiers and specifiers in Categorical Unification Grammar'. *Linguistics* **26**, 21–46.
- [Bouma & van Noord 94] Bouma, G. and G. van Noord (1994), 'Constraint-based categorial grammar'. Proceedings *ACL94*.
- [Bucalo 94] Bucalo, A. (1994), 'Modalities in Linear Logic weaker than the exponential "of course": algebraic and relational semantics'. *Journal of Logic, Language, and Information* **3**, 3, 211–232.
- [Buszkowski 84] Buszkowski, W. (1984), 'Fregean Grammar and residuated semigroups'. In G. Wechsung (ed.) *Frege Conference 1984*. Akademie-Verlag, Berlin.
- [Buszkowski 86] Buszkowski, W. (1986), 'Completeness results for Lambek syntactic calculus'. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, **32**, 13–28.
- [Buszkowski 87] Buszkowski, W. (1987), 'The logic of types'. In J.T. Srzednicki (ed.) *Initiatives in Logic*. Nijhoff, The Hague.
- [Buszkowski 88] Buszkowski, W. (1988), 'Generative power of categorial grammars'. In [Oehrle e.a. 88], 69–94.
- [Buszkowski e.a. 88] Buszkowski, W., W. Marciszewski and J. van Benthem (eds.) (1988), *Categorial Grammar*. John Benjamins, Amsterdam.
- [Buszkowski & Penn 90] Buszkowski, W. & G. Penn (1990), 'Categorial grammars determined from linguistic data by unification'. *Studia Logica* **49**, 431–454.
- [Calder e.a. 88] Calder, J., E. Klein and H. Zeevat (1988), 'Unification Categorial Grammar: a concise, extendable grammar for natural language processing'. Proceedings *COLING 1988*, Budapest, 83–86.
- [Carpenter 91] Carpenter, B. (1991), 'The generative power of categorial grammars and head-driven phrase structure grammars with lexical rules'. *Computational Linguistics*, **17**, 301–314.
- [Carpenter 92a] Carpenter, B. (1992), *Typed Feature Structures*. Cambridge.
- [Carpenter 92b] Carpenter, B. (1992), 'Categorial grammars, lexical rules, and the English predicative'. In R. Levine (ed.) *Formal Grammar: Theory and Implementation*. Cambridge.
- [Carpenter 94] Carpenter, B. (1994), 'Quantification and scoping: a deductive account'. *Proceedings 13th West Coast Conference on Formal Linguistics*. San Diego.
- [Carpenter 96] Carpenter, B. (1996), *Type-Logical Semantics*. MIT Press, Cambridge, MA.
- [Curry 61] Curry, H. (1961), 'Some logical aspects of grammatical structure'. In [Jakobson 61], 56–68.
- [Curry & Feys 58] Curry, H. & R. Feys (1958), *Combinatory Logic. Vol I*. Amsterdam.

- [Dalrymple e.a. 95] Dalrymple, M. J. Lamping, F. Pereira and V. Saraswat (1995), 'Linear Logic for Meaning Assembly'. In [Morrill & Oehrle], 75–93.
- [Dörre e.a. 94] Dörre, J., D. Gabbay and E. König (1994), 'Fibered semantics for feature-based grammar logic'. In Dörre (ed.) *Computational aspects of constraint-based linguistic description*. Esprit BRA Dyana-2 Deliverable R1.2.B. To appear in *Journal of Logic, Language, and Information*.
- [Dörre & Manandhar 95] Dörre, J. and S. Manandhar (1995), 'On constraint-based Lambek calculi'. In Blackburn and de Rijke (eds.) *Logic, Structures and Syntax*. Reidel, Dordrecht (to appear).
- [Došen 92] Došen, K. (1992), 'A brief survey of frames for the Lambek calculus'. *Zeitschr. f. math. Logik und Grundlagen d. Mathematik* **38**, 179–187.
- [Došen 89] Došen, K. (1988, 1989), 'Sequent systems and groupoid models'. *Studia Logica* **47**, 353–385 **48**, 41–65.
- [Došen & Schröder-Heister 93] Došen, K. & P. Schröder-Heister (eds.) (1993), *Substructural Logics*. Oxford.
- [Dowty 88] Dowty, D. (1988), 'Type-raising, functional composition, and non-constituent conjunction'. In [Oehrle e.a. 88], 153–197.
- [Dowty 91] Dowty, D. (1991), 'Towards a minimalist theory of syntactic structure'. In W. Sijtsma and A. van Horck (eds.) *Discontinuous Constituency*. De Gruyter, Berlin (to appear).
- [Dunn 93] Dunn, M. (1993), 'Partial Gaggles Applied to Logics With Restricted Structural Rules'. In [Došen & Schröder-Heister 93], 63–108.
- [Emms 93a] Emms, M. (1993), 'Parsing with polymorphism'. *Proceedings of the Sixth Conference of the European ACL*, Utrecht, 120–129.
- [Emms 93b] Emms, M. (1993) 'Some applications of categorial polymorphism'. In M. Moortgat (ed.) *Polymorphic Treatments*. Esprit BRA 6852 Dyana-2 Deliverable R1.3.A, 1–52.
- [Emms 94a] Emms, M. (1994), 'Extraction-covering extensions of the Lambek calculus are not CF'. In P. Dekker & M. Stokhof (eds.) *Proceedings of the Ninth Amsterdam Colloquium*, ILLC, Amsterdam, 269–286.
- [Emms 94b] Emms, M. (1994), 'Movement in polymorphic and labelled calculi'. In [Abrusci e.a. 94], 77–98.
- [Emms 94c] Emms, M. (1994), 'Completeness results for polymorphic Lambek calculus'. In M. Moortgat (ed.) *Lambek Calculus. Multimodal and Polymorphic Extensions*. Esprit BRA 6852 Dyana-2 Deliverable R1.1.B, 73–100.
- [Emms 95] Emms, M. (1995), 'An undecidability result for polymorphic Lambek calculus'. In M. Moortgat (ed.) *Logics of Structured Resources*. Esprit BRA 6852 Dyana-2 Deliverable R1.1.C, 59–77.
- [Emms & Leiss 93] Emms, M. & H. Leiss (1993) 'The Cut-Elimination theorem for the second order Lambek calculus'. In H. Leiss (ed.) *Categorial Parsing and Normalization*. Esprit BRA 6852 Dyana-2 Deliverable R1.1.A, 77–100.
- [Gabbay 94] Gabbay, D. (1994), *LDS — Labeled Deductive Systems*. Report MPI-I-94-223, Max-Planck-Institut für Informatik, Saarbrücken. (To appear with Oxford University Press.)

- [Gabbay & Kempson 92] Gabbay, D. & R. Kempson (1992) 'Natural language content: a truth-theoretic perspective'. In P. Dekker & M. Stokhof (eds.) *Proceedings Eighth Amsterdam Colloquium*, ILLC, Amsterdam, 173–198.
- [Geach 72] Geach, P. (1972), 'A program for syntax'. In D. Davidson and G. Harman (eds.) *Semantics of Natural Language*. Reidel, Dordrecht, 483–497. (Also in [Buszkowski e.a. 88]).
- [Gentzen 34] Gentzen, G. (1934), 'Untersuchungen über das logische Schliessen'. *Mathematische Zeitschrift* **39**, 176–210, 405–431.
- [Girard 87] Girard, J.-Y. (1987), 'Linear logic'. *Theoretical Computer Science* **50**, 1–102.
- [Girard 93] Girard, J.-Y. (1993), 'On the unity of logic'. *Annals of Pure and Applied Logic* **59**, 201–217.
- [Girard 95a] Girard, J.-Y. (1995), 'Geometry of interaction III: the general case'. In Girard, Lafont and Regnier (eds.) *Advances in Linear Logic*. Cambridge, 329–389.
- [Girard 95b] Girard, J.-Y., (1995), 'Light Linear Logic'. Ms LMD, Marseille.
- [Girard e.a. 89] Girard, J.-Y., P. Taylor, & Y. Lafont (1989), *Proofs and Types*. Cambridge Tracts in Theoretical Computer Science 7, Cambridge.
- [Hendriks 93] Hendriks, H. (1993), *Studied Flexibility. Categories and Types in Syntax and Semantics*. Ph.D. Dissertation, ILLC, Amsterdam.
- [Hendriks 94] Hendriks, H. (1994), 'Information packaging in a categorial perspective'. In E. Engdahl (ed.) *Integrating information structure into constraint-based and categorial approaches*. Esprit BRA 6852 Dyana-2 Deliverable R1.3.B, 89–116.
- [Hendriks 95] Hendriks, P. (1995), *Comparatives and Categorial Grammar*. Ph.D. Dissertation, Groningen.
- [Hepple 90] Hepple, M. (1990), *The Grammar and Processing of Order and Dependency*. Ph.D. Dissertation, Edinburgh.
- [Hepple 92] Hepple, M. (1992), 'Command and domain constraints in a categorial theory of binding'. *Proceedings Eighth Amsterdam Colloquium*, 253–270.
- [Hepple 94] Hepple, M. (1994), 'Labelled deduction and discontinuous constituency'. In [Abrusci e.a. 94], 123–150.
- [Hepple 95] Hepple, M. (1995), 'Hybrid Categorial Logics'. *Bulletin of the IGPL* **3**(2,3). Special issue on Deduction and Language (ed. R. Kempson), 343–355.
- [Hinrichs & Nakazawa 94] Hinrichs, E. and T. Nakazawa (1994), 'Linearizing AUXs in German verbal complexes'. In Nerbonne, Netter and Pollard (eds.) *German Grammar in HPSG*. CSLI Lecture Notes, Stanford, pp 11–38.
- [Hodas & Miller 94] Hodas, J.S. & D. Miller (1994), 'Logic programming in a fragment of Intuitionistic Linear Logic'. *Information and Computation* **110**, 327–365.
- [Jacobson 87] Jacobson, P. (1987), 'Phrase structure, grammatical relations, and discontinuous constituents'. In G.J. Huck and A.E. Ojeda (eds.) *Syntax and Semantics 20: Discontinuous Constituency*. Academic Press, New York. 27–69.
- [Jakobson 61] Jakobson, R. (ed.) (1961), *Structure of Language and Its Mathematical Aspects*. Proceedings of the Twelfth Symposium in Applied Mathematics. Providence, Rhode Island.

- [Joshi & Kulick 95] Joshi, A. & S. Kulick (1995), 'Partial proof trees as building blocks for a categorial grammar'. In [Morrill & Oehrle], 138–149.
- [Kanazawa 92] Kanazawa, M. (1992), 'The Lambek calculus enriched with additional connectives'. *Journal of Logic, Language, and Information* **1**, 141–171.
- [Kanazawa 94] Kanazawa, M. (1994), *Learnable Classes of Categorical Grammars*. Ph.D. Dissertation. Stanford.
- [Kandulski 88] Kandulski, W. (1988) 'The non-associative Lambek calculus'. In [Buszkowski e.a. 88], 141–151.
- [Keenan & Faltz 85] Keenan, E.L. & L. Faltz (1985) *Boolean Semantics for Natural Language*. Reidel, Dordrecht.
- [Kołowska 95] Kołowska-Gawiejnowics, M. (1995), 'Powerset residuated algebras and generalized Lambek calculus'. Report 36/1995, Faculty of Mathematics and Computer Science, Adam Mickiewicz University, Poznań. To appear in *Mathematical Logic Quarterly*.
- [König 91] König, E. (1991) 'Parsing as natural deduction'. *Proceedings of the 27th Annual Meeting of the ACL*, Vancouver, 272–279.
- [König 95] König, E. (1995), 'LexGram — a practical categorial grammar formalism'. *Proceedings Computational Logic for Natural Language Processing*. Edinburgh.
- [Kraak 95] Kraak, E. (1995), 'French object clitics: a multimodal analysis'. In [Morrill & Oehrle], 166–180.
- [Kracht 93] Kracht, M. (1993), 'Power and weakness of the modal Display Calculus'. Ms Freie Universität Berlin.
- [Kurtonina 95] Kurtonina, N. (1995), *Frames and Labels. A Modal Analysis of Categorical Inference*. Ph.D. Dissertation, OTS Utrecht, ILLC Amsterdam.
- [Kurtonina & Moortgat 95] Kurtonina, N. & M. Moortgat (1995), 'Structural Control'. In M. Moortgat (ed.) *Logics of Structured Resources*. Esprit BRA 6852 Dyana-2 Deliverable R1.1.C. (To appear in P. Blackburn & M. de Rijke (eds.) *Logic, Structures and Syntax*. Reidel, Dordrecht.)
- [Lambek 58] Lambek, J. (1958), 'The Mathematics of Sentence Structure', *American Mathematical Monthly* **65**, 154–170.
- [Lambek 61] Lambek, J. (1961), 'On the calculus of syntactic types'. In [Jakobson 61].
- [Lambek 68] Lambek, J. (1968), 'Deductive systems and categories. I', *J. Math. Systems Theory* **2**, 278–318.
- [Lambek 88] Lambek, J. (1988), 'Categorial and categorial grammar'. In [Oehrle e.a. 88], 297–317.
- [Lambek 93a] Lambek, J. (1993), 'Logic without structural rules. (Another look at Cut Elimination)'. In [Došen & Schröder-Heister 93], 179–206.
- [Lambek 93b] Lambek, J. (1993), 'From categorial grammar to bilinear logic'. In [Došen & Schröder-Heister 93], 207–238.
- [Lecomte & Retoré 95] 'Pomset logic as an alternative categorial grammar'. In [Morrill & Oehrle], 181–196.
- [Lewis 72] Lewis, D. (1972), 'General semantics'. In D. Davidson and G. Harman (eds.) *Semantics of Natural Language*. Reidel, Dordrecht, 169–218.

- [Lincoln e.a. 95] Lincoln, P., A. Scedrov and N. Shankar (1995) 'Decision problems for second-order Linear Logic'. *Proceedings of the Tenth Annual IEEE Symposium on Logic in Computer Science*.
- [Lyons 68] Lyons, J. (1968), *Theoretical Linguistics*. Cambridge.
- [Miller 92] Miller, Ph. (1992), *Clitics and Constituents in Phrase Structure Grammar*. Garland, New York.
- [Moortgat 88] Moortgat, M. (1988), *Categorial Investigations. Logical and Linguistic Aspects of the Lambek Calculus*. Foris, Dordrecht.
- [Moortgat 91] Moortgat, M. (1991), 'Generalized quantification and discontinuous type constructors'. (To appear in W. Sijtsma & A. van Horck (eds.) *Discontinuous Constituency*, De Gruyter, Berlin.)
- [Moortgat 92] Moortgat, M. (1992), 'Labelled Deductive Systems for categorial theorem proving'. *Proceedings Eighth Amsterdam Colloquium*, ILLC, Amsterdam, 403–424.
- [Moortgat 95] Moortgat, M. (1995), 'Multimodal linguistic inference'. *Bulletin of the IGPL* **3**(2,3). Special issue on Deduction and Language (ed. R. Kempson), 371–401. (To appear in *Journal of Logic, Language, and Information*).
- [Moortgat & Morrill 91] Moortgat, M. and G. Morrill (1991), 'Heads and phrases. Type calculus for dependency and constituent structure'. Ms OTS Utrecht.
- [Moortgat & Oehrle 93] Moortgat, M. and R. Oehrle (1993), *Logical parameters and linguistic variation. Lecture notes on categorial grammar*. 5th European Summer School in Logic, Language and Information. Lisbon.
- [Moortgat & Oehrle 94] Moortgat, M. and R.T. Oehrle (1994), 'Adjacency, dependency and order'. In P. Dekker and M. Stokhof (eds.) *Proceedings Ninth Amsterdam Colloquium*, ILLC, Amsterdam, 447–466.
- [Morrill 90a] Morrill, G. (1990), 'Intensionality and Boundedness'. *Linguistics & Philosophy* **13**, 699–726.
- [Morrill 90b] Morrill, G. (1990), 'Grammar and logical types'. In M. Stokhof and L. Torenvliet (eds.) *Proceedings of the Seventh Amsterdam Colloquium*, ILLC, Amsterdam, 429–450.
- [Morrill 94a] Morrill, G. (1994), *Type Logical Grammar. Categorial Logic of Signs*. Kluwer, Dordrecht.
- [Morrill 94b] Morrill, G. (1994), 'Structural facilitation and structural inhibition'. In [Abrusci e.a. 94], 183–210.
- [Morrill 95a] Morrill, G. (1995), 'Discontinuity in Categorial Grammar'. *Linguistics & Philosophy* **18**, 175–219.
- [Morrill 95b] Morrill, G. (1995), 'Higher-order Linear Logic programming of categorial deduction'. *Proceedings of the European Chapter of the Association for Computational Linguistics*, Dublin.
- [Morrill 95c] Morrill, G. (1995), 'Clausal Proofs and Discontinuity'. *Bulletin of the IGPL* **3**(2,3). Special issue on Deduction and Language (ed. R. Kempson), 403–427.
- [Morrill & Oehrle] Morrill, G. and R.T. Oehrle (1995), *Formal Grammar*. Proceedings of the Conference of the European Summer School in Logic, Language and Information. Barcelona.
- [Muskens xx] Muskens, R. (19xx), 'Categorial grammar and Discourse Representation Theory'.

- [Oehrle 88] Oehrle, R.T. (1988), 'Multi-dimensional compositional functions as a basis for grammatical analysis'. In [Oehrle e.a. 88], 349–389.
- [Oehrle 95] Oehrle, R.T. (1995), 'Term-labeled categorial type systems', *Linguistics & Philosophy* **17**, 633–678.
- [Oehrle e.a. 88] Oehrle, R.T., E. Bach & D. Wheeler (eds.) (1988), *Categorial Grammars and Natural Language Structures*. Reidel, Dordrecht.
- [Partee & Rooth 83] Partee, B. & M. Rooth (1983), 'Generalized conjunction and type ambiguity'. In Bäuerle e.a. (eds.) *Meaning, Use, and Interpretation of Language*, De Gruyter, Berlin, 361–383.
- [Pollard 84] Pollard, C. (1984), *Head Grammars, Generalized Phrase Structure Grammars, and Natural Language*. Ph.D. Dissertation, Stanford.
- [Pollard & Sag 94] Pollard, C. & I. Sag (1994), *Head-Driven Phrase Structure Grammar*. Chicago.
- [Pentus 93] Pentus, M. (1993), 'Lambek grammars are context free'. *Proceedings Eighth Annual IEEE Symposium on Logic in Computer Science*, Montreal.
- [Pentus 94] Pentus, M. (1994), 'Language Completeness of the Lambek Calculus'. *Proceedings 9th Annual IEEE Symposium on Logic in Computer Science*, Paris.
- [Reape 89] Reape, M. (1989), 'A logical treatment of semi-free word order and bounded discontinuous constituency'. In *Proceedings of the Fourth Conference of the European Chapter of the Association for Computational Linguistics*, Manchester, 103–115.
- [Restall 93] Restall, G. (1993), 'Simplified semantics for Relevant Logics (and some of their rivals)'. *Journal of Philosophical Logic* **22**, 279–303.
- [Restall 94] Restall, G. (1994), 'A useful substructural logic'. *Bulletin of the IGPL* **2**, 137–148.
- [Roorda 91] Roorda, D. (1991), *Resource Logics. Proof-Theoretical Investigations*. Ph.D. Dissertation, Amsterdam.
- [Sag e.a. 85] Sag, I., G. Gazdar, Th. Wasow and S. Weisler (1985), 'Coordination and how to distinguish categories'. *Natural Language & Linguistic Theory* **3**, 117–171.
- [Sanchez 95] Sánchez Valencia, V. (1995), 'Natural logic: parsing driven inference'. To appear in *Linguistic Analysis*.
- [Shieber 86] Shieber, S.M. (1986), *An Introduction to Unification-Based Approaches to Grammar*. CSLI Lecture Notes, Chicago.
- [Shieber 92] Shieber, S.M. (1992), *Constraint-Based Grammar Formalisms. Parsing and Type Inference for Natural and Computer Languages*. MIT Press, Cambridge, MA.
- [Schmerling 83] Schmerling, S. (1983), 'A new theory of English auxiliaries'. In F. Heny & B. Richards (eds.) *Linguistic Categories. Auxiliaries and Related Puzzles*, Vol II, Reidel, Dordrecht, 1–53.
- [Steedman 84] Steedman, M. (1984), 'A categorial theory of intersecting dependencies in Dutch infinitival complements'. In De Geest & Putseys (eds.) *Proceedings of the International Conference on Complementation*, Foris, Dordrecht, 215–226.
- [Steedman 85] Steedman, M. (1985), 'Dependency and coordination in the grammar of Dutch and English'. *Language* **61**, 523–568.
- [Steedman 87] Steedman, M. (1987), 'Combinatory Grammars and parasitic gaps'. *Natural Language & Linguistic Theory* **5**, 403–439.

- [Steedman 88] Steedman, M. (1988), ‘Combinators and grammars’. In [Oehrle e.a. 88], 417–442.
- [Steedman 91] Steedman, M. (1991), ‘Structure and Intonation’. *Language* **68**, 260–296.
- [Steedman 93] Steedman, M. (1993), ‘Categorial Grammar. Tutorial Overview’. *Lingua* **90**, 221–258.
- [Steedman 94] Steedman, M. (1994), ‘The grammar of intonation and focus’. In Dekker, P. & M. Stokhof (eds.) *Proceedings 9th Amsterdam Colloquium*, 17–33. ILLC, Amsterdam.
- [Szabolcsi 87] Szabolcsi, A. (1987), ‘On Combinatory Categorial Grammar’. *Proceedings of the Symposium on Logic and Language, Debrecen*, 151–162. Budapest.
- [Trautwein 95] Trautwein, M. (1995), *Computational Pitfalls in Tractable Grammar Formalisms*. Ph.D. Thesis, ILLC, Amsterdam.
- [Troelstra 92] Troelstra, A.S. (1992), *Lectures on Linear Logic*. CSLI Lecture Notes. Stanford.
- [Uszkoreit 86] Uszkoreit, H. (1986), ‘Categorial Unification Grammar’. *Proceedings COLING 1986, Bonn*, pp. 187–194.
- [Venema 93] Venema, Y. (1993), ‘Meeting strength in substructural logics’. UU Logic Preprint. To appear in *Studia Logica*.
- [Venema 94] Venema, Y. (1994), ‘Tree models and (labeled) categorial grammar’. In P. Dekker and M. Stokhof (eds.) *Proceedings Ninth Amsterdam Colloquium*, ILLC, Amsterdam, 703–722.
- [Versmissen 93] Versmissen, K. (1993), ‘Categorial grammar, modalities and algebraic semantics’. *Proceedings EACL93*, 377–383.
- [Versmissen 96] Versmissen, K. (1996), *Grammatical Composition. Modes, Models and Modalities*. Ph.D. Dissertation. OTS Utrecht.
- [Wallen 90] Wallen, L.A. (1990), *Automated Deduction in Nonclassical Logics*. MIT Press, Cambridge, MA.
- [Wansing 92a] Wansing, H. (1992a), ‘Sequent calculi for normal modal propositional logics’. ILLC Report LP–92–12.
- [Wansing 92b] Wansing, H. (1992b), ‘Formulas-as-types for a hierarchy of sublogics of intuitionistic propositional logic’. In D. Pearce & H. Wansing (eds.) *Non-classical Logics and Information Processing*. Springer Lecture Notes in AI 619. Berlin.
- [Wansing 92c] Wansing, H. (1992c), *The Logic of Information Structures*. Ph.D. Dissertation. Berlin.
- [Zielonka 89] Zielonka, W. (1989), ‘A simple and general method of solving the finite axiomatizability problems for Lambek’s syntactic calculi’. *Studia Logica* **48**, 35–39.
- [Zwarts 86] Zwarts, F. (1986), *Categoriale grammatica en algebraïsche semantiek*. Ph.D. Dissertation, Groningen.