Categorial Type Logics

Michael Moortgat
Research Institute for Language and Speech
(OTS, Utrecht)

Chapter Two in J. van Benthem and A. ter Meulen (eds.)
Handbook of Logic and Language. Elsevier, to appear.

Contents

Definitions

Definitions (continued)

Propositions

Examples

Examples (continued)

Other Handbook Chapters

Les quantités du langage et leurs rapports sont regulièrement 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 Saussure[1]

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: $\Diamond, \Box^{\downarrow}, /, \bullet, \backslash$. 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

[1] N10 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].

language semantics studied from a type-logical perspective. Chapter Twelve of this handbook discusses categorial grammar logics form 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 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 $B\backslash A$ 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, $B\backslash A$ 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 $\backslash$ 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 $B, B\backslash A \vdash A$.

$$\text{from } \Gamma \vdash A/B \text{ and } \Delta \vdash B, \text{ infer } \Gamma, \Delta \vdash A \qquad \text{from } \Gamma \vdash B \text{ and } \Delta \vdash B\backslash A, \text{ infer } \Gamma, \Delta \vdash A \tag{1}$$

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], [\E] for the 'elimination' of the implication connectives.

$$\dfrac{\text{Kazimierz}\quad \dfrac{\text{talks}\quad \dfrac{\text{to}\quad \dfrac{\text{the}\quad \text{mathematician}}{(np/n)\qquad n}/\text{E}}{(pp/np)\qquad np}/\text{E}}{((np\backslash s)/pp)\qquad pp}/\text{E}}{np\qquad (np\backslash s)}\backslash\text{E}\qquad s \tag{2}$$

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 perspectiveΓ In other words: given information about the categorization of a composite structure, what conclusions could we draw about the categorization of its partsΓ Suppose we want to find out whether a structure Γ can be appropriately categorized as A/B. Given the interpretation we had in mind for the implication /, such a conclusion would be justified if we could show that Γ in construction with an arbitrary expression of type B can be categorized as an expression of type A. Similarly, from the grammaticality judgement that B in construction with Γ is of type A, we can conclude that Γ itself is of type $B\backslash A$. The inference patterns (3), introduced in [Lambek 58], tell us how to *prove* formulas A/B or $B\backslash A$, just as the (1) inferences told us how to *use* these implications.

$$\text{from } \Gamma, B \vdash A, \text{ infer } \Gamma \vdash A/B \qquad \text{from } B, \Gamma \vdash A, \text{ infer } \Gamma \vdash B\backslash A \tag{3}$$

In order to see where this type of 'deconstructive' reasoning comes into play, consider the relative clause example 'the mathematician whom Kazimierz talks to'. There is one new lexical item in this example: the relative pronoun *whom*. This item is categorized as incomplete: on the right, it wants to enter into composition with the relative clause body — an expression which we would like to assign to the category s/np.

$$\dfrac{\dfrac{\text{the}}{(np/n)}\quad \dfrac{\dfrac{\text{mathematician}}{n}\quad \dfrac{\dfrac{\text{whom}}{((n\backslash n)/(s/np))}\quad \dfrac{\dfrac{\text{Kazimierz}}{np}\quad \dfrac{\text{talks}\quad \dfrac{\text{to}}{(pp/np)\qquad \overline{np}}/\text{E}}{((np\backslash s)/pp)\qquad pp}/\text{E}}{(np\backslash s)}\backslash\text{E}}{s}/\text{I}}{(s/np)}/\text{E}}{(n\backslash n)}\backslash\text{E}}{n}/\text{E}}{np} \tag{4}$$

In order to show that 'Kazimierz talks to' is indeed of type s/np, we make a *hypothetical* assumption, and suppose we have an arbitrary np expression. With the aid of this hypothetical assumption, we derive s for 'Kazimierz talks to np', using the familiar Modus Ponens steps of inference. At the point where we have derived s, we withdraw the hypothetical np assumption, and conclude that 'Kazimierz talks to' can be categorized as s/np. This

step is labeled [/I], for the 'introduction' of the implication connective, and the withdrawn assumption is marked by overlining.

The relation between the *wh* pronoun and the hypothetical *np* position which it preempts is often described metaphorically in terms of 'movement'. Notice that in our deductive setting we achieve the effects of 'movement' without adding anything to the theory of grammatical composition: there is no need for abstract syntactic place-holders (such as the 'empty' trace categories of Chomskyan syntax, or the $\mathbf{he}_i$ syntactic variables of Montague's PTQ), nor for extra combination schemata beyond Modus Ponens. The similarity between the Natural Deduction graphs and phrase structure trees, in other words, is misleading: what we have represented graphically are the steps in a deductive process — not to be confused with the construction of a syntactic tree.

In the above, we have talked about the *form* dimension of grammatical composition: about putting together linguistic resources into well-formed structural configurations. But a key point of the categorial approach is that one can simultaneously consider the types/categories, and hence grammatical composition, in the *meaning* dimension. From the semantic perspective, one fixes the kind of meaning objects one wants for the basic types that categorize complete expressions, and then interprets objects of types A/B, $B\backslash A$ as *functions* from A type objects to B type objects. Structural composition by means of Modus Ponens can then be naturally correlated with functional application, and Hypothetical Reasoning with functional abstraction in the semantic dimension. Composition of linguistic form and meaning composition thus become aspects of one and the same process of grammatical inference.

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, ...*) 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.

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

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 §5.1 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 $A \to B$ 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 $/, \backslash$ with respect to derivability. Prop 2.3 states the essential soundness and completeness result with respect to the frame semantics. (We write '$\mathcal{L} \vdash A \to B$' for '$A \to B$ is provable in logic $\mathcal{L}$').

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

$$\text{(REFL)} \quad A \to A \qquad \text{(TRANS)} \quad \text{from } A \to B \text{ and } B \to C, \text{ infer } A \to C$$

$$\text{(RES)} \qquad A \to C/B \quad \textit{iff} \quad A \bullet B \to C \quad \textit{iff} \quad B \to A \backslash C$$

PROPOSITION 2.3. ([Došen 92]). $\mathbf{NL} \vdash A \to B$ 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 $A \to B$. 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 $\mathbf{NL} \vdash A \to \phi$. Due to the Truth Lemma we have that if $\mathbf{NL} \nvdash A \to B$, 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 $\mathcal{M}_K = \langle W_K, R^3_K, v_K \rangle$, where

(i) W_K is the set of formulae $\mathcal{F}$
(ii) $R^3_K(A, B, C)$ iff $\mathbf{NL} \vdash A \to B \bullet C$
(iii) $A \in v_K(p)$ iff $\mathbf{NL} \vdash A \to p$

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 ($\forall x, y, z, u \in W$).

(ASS)	$(A \bullet B) \bullet C \longleftrightarrow A \bullet (B \bullet C)$	$\exists t.Rtxy \ \& \ Rutz \Leftrightarrow \exists v.Rvyz \ \& \ Ruxv$
(COMM)	$A \bullet B \rightarrow B \bullet A$	$Rxyz \Leftrightarrow Rxzy$

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.

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

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 **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 **L** with respect to the free semigroup models, there is an *incompleteness* result for **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 categorial 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 $\textsc{lex}(L)$. 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* categorial 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 $\textsc{lex}(L)$, 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 $x_1 \ldots x_n \in V_L^+$, provided there are lexical type specifications $A_i \in f(x_i)$ such that we can deduce B from $\circ(A_1, \ldots, A_n)$ in the general type-logic $\mathcal{L}$. By $\circ(A_1, \ldots, A_n)$ we mean any of the possible products of the formulas $A_1, \ldots, A_n$, in that order.

CATEGORICAL COMBINATORS AND CCG. To round off the discussion of the axiomatic presentation, we present the logics **NL, L, NLP, 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.

$$1_A : A \to A \qquad \frac{f : A \to B \quad g : B \to C}{g \circ f : A \to C}$$

$$\frac{f : A \bullet B \to C}{\beta_{A,B,C}(f) : A \to C/B} \qquad \frac{f : A \bullet B \to C}{\gamma_{A,B,C}(f) : B \to A\backslash C}$$

$$\frac{g : A \to C/B}{\beta^{-1}_{A,B,C}(g) : A \bullet B \to C} \qquad \frac{g : B \to A\backslash C}{\gamma^{-1}_{A,B,C}(g) : A \bullet B \to C}$$

$$\alpha_{A,B,C} : A \bullet (B \bullet C) \longleftrightarrow (A \bullet B) \bullet C : \alpha^{-1}_{A,B,C}$$
$$\pi_{A,B} : A \bullet B \to B \bullet A$$

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 $\beta^{-1}(1_{A/B})$.

$$\frac{1_{A/B} : A/B \to A/B}{\beta^{-1}(1_{A/B}) : A/B \bullet B \to A} \qquad \frac{\mathsf{RA} : A/B \bullet B \to A}{\gamma(\mathsf{RA}) : B \to (A/B)\backslash A}$$

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 Categorial 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 $A \to B$ and $B \to C$ into a transition $A \to C$. A type transition $A \to C$, 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 $A \to B$ with $A, B \in \mathcal{F}$. In the Gentzen presentation, the derivability relation is stated to hold between a *term* $\mathcal{S}$ (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 $\mathcal{S} ::= \mathcal{F} \mid (\mathcal{S}, \mathcal{S})$. 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 $f : A \to B$ there is a Gentzen proof of $A \Rightarrow B$, and for every proof of a sequent $\Gamma \Rightarrow B$ there is an arrow $f : \Gamma^\circ \to B$.

DEFINITION 2.12. **NL**: Gentzen presentation. Sequents $\mathcal{S} \Rightarrow \mathcal{F}$ where $\mathcal{S} ::= \mathcal{F} \mid (\mathcal{S}, \mathcal{S})$. 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.)

$$[\mathrm{Ax}]\frac{}{A \Rightarrow A} \qquad \frac{\Delta \Rightarrow A \quad \Gamma[A] \Rightarrow C}{\Gamma[\Delta] \Rightarrow C}[\mathrm{Cut}]$$

$$[/\mathrm{R}]\frac{(\Gamma, B) \Rightarrow A}{\Gamma \Rightarrow A/B} \qquad \frac{\Delta \Rightarrow B \quad \Gamma[A] \Rightarrow C}{\Gamma[(A/B, \Delta)] \Rightarrow C}[/\mathrm{L}]$$

$$[\backslash\mathrm{R}]\frac{(B, \Gamma) \Rightarrow A}{\Gamma \Rightarrow B\backslash A} \qquad \frac{\Delta \Rightarrow B \quad \Gamma[A] \Rightarrow C}{\Gamma[(\Delta, B\backslash A)] \Rightarrow C}[\backslash\mathrm{L}]$$

$$[\bullet\mathrm{L}]\frac{\Gamma[(A, B)] \Rightarrow C}{\Gamma[A \bullet B] \Rightarrow C} \qquad \frac{\Gamma \Rightarrow A \quad \Delta \Rightarrow B}{(\Gamma, \Delta) \Rightarrow A \bullet B}[\bullet\mathrm{R}]$$

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 $A \to B$ where A and B are constructed out of formula variables $p_1, \ldots, p_n$ and the logical connective $\bullet$. For corresponding structure variables $\Delta_1, \ldots, \Delta_n$ 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 $A \to B$ translates as the Gentzen rule (7):

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

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.)

$$\frac{\Gamma[(\Delta_2, \Delta_1)] \Rightarrow A}{\Gamma[(\Delta_1, \Delta_2)] \Rightarrow A}[\mathrm{P}] \qquad \frac{\Gamma[((\Delta_1, \Delta_2), \Delta_3)] \Rightarrow A}{\Gamma[(\Delta_1, (\Delta_2, \Delta_3))] \Rightarrow A}[\mathrm{A}] \tag{8}$$

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. Antecendent terms then take the form of sequences of formulae $A_1, \ldots, A_n$ 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 $\mathcal{S} \Rightarrow \mathcal{F}$ where $\mathcal{S} ::= \mathcal{F} \mid \mathcal{F}, \mathcal{S}$. **L**: implicit Associativity, interpreting $\mathcal{S}$ as a sequence. **LP**: implicit Associativity+Permutation, interpreting $\mathcal{S}$ as a multiset. (The context variables Γ, Γ' can be empty.)

$$[\text{Ax}]\frac{}{A \Rightarrow A} \qquad \frac{\Delta \Rightarrow A \quad \Gamma, A, \Gamma' \Rightarrow C}{\Gamma, \Delta, \Gamma' \Rightarrow C}[\text{Cut}]$$

$$[/\text{R}]\frac{\Delta, B \Rightarrow A}{\Delta \Rightarrow A/B} \qquad \frac{\Delta \Rightarrow B \qquad \Gamma, A, \Gamma' \Rightarrow C}{\Gamma, A/B, \Delta, \Gamma' \Rightarrow C}[/\text{L}]$$

$$[\backslash\text{R}]\frac{B, \Delta \Rightarrow A}{\Delta \Rightarrow B\backslash A} \qquad \frac{\Delta \Rightarrow B \qquad \Gamma, A, \Gamma' \Rightarrow C}{\Gamma, \Delta, B\backslash A, \Gamma' \Rightarrow C}[\backslash\text{L}]$$

$$[\bullet\text{L}]\frac{\Gamma, A, B, \Gamma' \Rightarrow C}{\Gamma, A \bullet B, \Gamma' \Rightarrow C} \qquad \frac{\Delta \Rightarrow A \qquad \Delta' \Rightarrow B}{\Delta, \Delta' \Rightarrow A \bullet B}[\bullet\text{R}]$$

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.

$$\dfrac{\dfrac{\Delta'' \Rightarrow A'' \quad \Delta, A', \Delta' \Rightarrow A}{\Delta, A'/A'', \Delta'', \Delta' \Rightarrow A}\,/L \qquad \Gamma, A, \Gamma' \Rightarrow B}{\Gamma, \Delta, A'/A'', \Delta'', \Delta', \Gamma' \Rightarrow B}\,\text{Cut} \quad \rightsquigarrow \quad \dfrac{\Delta'' \Rightarrow A'' \qquad \dfrac{\Delta, A', \Delta' \Rightarrow A \quad \Gamma, A, \Gamma' \Rightarrow B}{\Gamma, \Delta, A', \Delta', \Gamma' \Rightarrow B}\,\text{Cut}}{\Gamma, \Delta, A'/A'', \Delta'', \Delta', \Gamma' \Rightarrow B}\,/L$$

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

$$\dfrac{\dfrac{\Delta, A'' \Rightarrow A'}{\Delta \Rightarrow A'/A''}\,/R \quad \dfrac{\Delta' \Rightarrow A'' \quad \Gamma, A', \Gamma' \Rightarrow B}{\Gamma, A'/A'', \Delta', \Gamma' \Rightarrow B}\,/L}{\Gamma, \Delta, \Delta', \Gamma' \Rightarrow B}\,\text{Cut} \quad \rightsquigarrow \quad \dfrac{\Delta' \Rightarrow A'' \qquad \dfrac{\Delta, A'' \Rightarrow A' \quad \Gamma, A', \Gamma' \Rightarrow B}{\Gamma, \Delta, A'', \Gamma' \Rightarrow B}\,\text{Cut}}{\Gamma, \Delta, \Delta', \Gamma' \Rightarrow B}\,\text{Cut}$$

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 $A\backslash B$ (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 [\L] can be found below.

$$\dfrac{\Delta \Rightarrow B \qquad \dfrac{B \Rightarrow B \quad A \Rightarrow A}{B, B\backslash A \Rightarrow A}\,\backslash L}{\Delta, B\backslash A \Rightarrow A}\,[\text{Cut}] \qquad \Gamma, A, \Gamma' \Rightarrow C}{\Gamma, \Delta, B\backslash A, \Gamma' \Rightarrow C}\,[\text{Cut}] \quad \rightsquigarrow \quad \dfrac{\Delta \Rightarrow B \quad \Gamma, A, \Gamma' \Rightarrow C}{\Gamma, \Delta, B\backslash A, \Gamma' \Rightarrow C}\,\backslash L \tag{10}$$

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 §7.1 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).

$$A \vdash A$$

$$[/\mathrm{I}]\frac{(\Gamma, B) \vdash A}{\Gamma \vdash A/B} \qquad \frac{\Gamma \vdash A/B \quad \Delta \vdash B}{(\Gamma, \Delta) \vdash A}[/\mathrm{E}]$$

$$[\backslash\mathrm{I}]\frac{(B, \Gamma) \vdash A}{\Gamma \vdash B\backslash A} \qquad \frac{\Gamma \vdash B \quad \Delta \vdash B\backslash A}{(\Gamma, \Delta) \vdash A}[\backslash\mathrm{E}]$$

$$[\bullet\mathrm{I}]\frac{\Gamma \vdash A \quad \Delta \vdash B}{(\Gamma, \Delta) \vdash A \bullet B} \qquad \frac{\Delta \vdash A \bullet B \quad \Gamma[(A, B)] \vdash C}{\Gamma[\Delta] \vdash C}[\bullet\mathrm{E}]$$

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

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 ($\bullet$,\) 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 \, (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. $(s/(np\backslash s))\backslash s \to np\backslash s$) and Value Raising (e.g. $np/n \to (s/(np\backslash s))/n$), 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 $(B/A)\backslash B$.)

$$A \to A^B \qquad (A^B)^B \to A^B \qquad \text{from } A \to C, \text{ infer } A^B \to C^B \tag{11}$$

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 **b** for the left-division variant of principle (7) in Prop 2.18, and use X as an abbreviation for $A \bullet ((A \backslash (A \bullet B)) \bullet ((A \bullet B) \backslash ((A \bullet B) \bullet C)))$.)

$$\dfrac{\dfrac{\dfrac{\mathbf{b} : A\backslash B \to (C\backslash A)\backslash(C\backslash B)}{\gamma^{-1}(\mathbf{b}) : C\backslash A \bullet A\backslash B \to C\backslash B}}{(\dagger)\ \gamma^{-1}(\gamma^{-1}(\mathbf{b})) : C \bullet (C\backslash A \bullet A\backslash B) \to B}}{}$$

$$\dfrac{\dfrac{A \to A \qquad \dfrac{(2) : B \to A\backslash(A \bullet B) \quad (2) : C \to (A \bullet B)\backslash((A \bullet B) \bullet C)}{(3) : B \bullet C \to (A\backslash(A \bullet B)) \bullet ((A \bullet B)\backslash((A \bullet B) \bullet C))}}{(3) : A \bullet (B \bullet C) \to X} \qquad (\dagger) : X \to (A \bullet B) \bullet C}{A \bullet (B \bullet C) \to (A \bullet B) \bullet C}$$

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]). The free availability of restructuring makes it possible to give alternative constituent analyses for expressions that would count as structurally unambiguous under rigid constituency assumptions, such as embodied by **NL**.

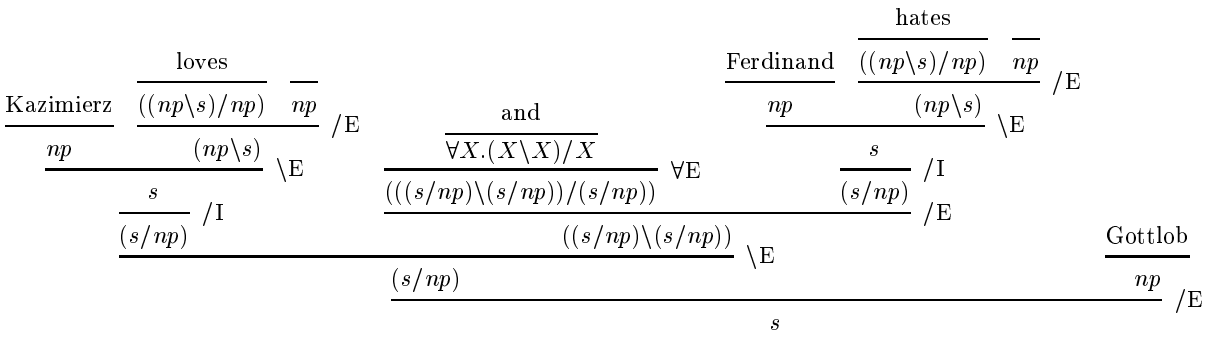
EXAMPLE 2.20. Restructuring: subject-(verb-object) versus (subject-verb)-object analysis. Derivation in CCG tree format, in terms of the combinators Lifting, Composition, and Application. One can see the CCG trees as concise representations of combinator proofs in the sense of Def 2.9, given as compositions of 'primitive' CCG arrows.

$$\dfrac{\dfrac{\text{Mary}}{np} \qquad \dfrac{\dfrac{\text{cooked}}{(np\backslash s)/np} \qquad \dfrac{\text{the beans}}{np}\ \mathbf{app}}{np\backslash s}\ \mathbf{app}}{s}$$

$$\dfrac{\dfrac{\dfrac{\text{Mary}}{np}}{s/(np\backslash s)}\ \mathbf{lift} \qquad \dfrac{\text{cooked}}{(np\backslash s)/np}\ \mathbf{comp} \qquad \dfrac{\text{the beans}}{np}}{s/np}\ \mathbf{app}}{s}$$

Coordination phenomena provide crucial motivation for associative resource management and the non-standard constituent analyses that come with it, cf. [Steedman 85, Dowty 88, Zwarts 86] for the original argumentation. On the assumption that coordination joins expressions of like category, theories of rigid constituency run into problems with cases of so-called non-constituent coordination, such as the Right Node Raising example below. With an associative theory of grammatical composition, non-constituent coordination can be reduced to standard coordination of phrases of like type. As will become clear in §3, the interpretation produced for the s/np instantiation of the coordination type is the appropriate one for a theory of generalized conjoinability such as [Keenan & Faltz 85, Partee & Rooth 83].

EXAMPLE 2.21. Conjunction of non-constituents. Natural Deduction format. See §5.2 for the type schema for 'and'.



Other types of argumentation for flexible constituency have been based on *processing* considerations (an associative regime can produce an *incremental* left-to-right analysis of a sentence cf. [Ades & Steedman 82]), or *intonational structure* (distinct prosodic phrasing realizing alternative information packaging for the same truth conditional content, cf. [Steedman 91]).

Unfortunately, the strength of **L** is at the same time its weakness. Associative resource management *globally* destroys discrimination for constituency, not just where one would like to see a relaxation of structure sensitivity. Standard constituent analyses provide the proper basis for the characterization of domains of locality: in an associative setting, the constituent information is lost. As examples of the resulting overgeneration one can cite violations of the Coordinate Structure Constraint such as (12). The type assignment $(n\backslash n)/(s/np)$ to the relative pronoun requires 'Gottlob admired Kazimierz and Jim detested' to be of type s/np. With an instantiation $(s\backslash s)/s$ for the conjunction, and an associative regime of composition, there is nothing that can stop the 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 ($np\backslash iv$)
'(that Mary) the hippos wants feed' (= that M. wants to feed the hippos)
$vp/iv, np\backslash iv \Rightarrow np\backslash vp$ (Mixed Composition)
$A/B \rightarrow (C\backslash A)/(C\backslash B)$ (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 $A_1, \ldots, A_n \Rightarrow B$ we now consider annotated sequents $x_1 : A_1, \ldots, x_n : A_n \Rightarrow t : B$ 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

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

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 $A, B \in \mathcal{F}$:

$$\mathcal{T}^A ::= \mathcal{V}^A \mid \mathcal{T}^{B \rightarrow A}(\mathcal{T}^B) \mid (\mathcal{T}^{A \circ B})_0 \mid (\mathcal{T}^{B \circ A})_1$$

$$\mathcal{T}^{A\to B} ::= \lambda\mathcal{V}^A\mathcal{T}^B \qquad \mathcal{T}^{A\circ B} ::= \langle\mathcal{T}^A, \mathcal{T}^B\rangle$$

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 $\Lambda(\mathbf{LP})$ 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 $x_1 : A_1, \ldots, x_n : A_n \Rightarrow t : B$, the antecedent x_i are distinct. For the implication $\to$, 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.

$$\frac{}{x : A \Rightarrow x : A}\ (\text{Ax}) \qquad \frac{\Gamma \Rightarrow t : A \quad x : A, \Delta \Rightarrow u : B}{\Gamma, \Delta \Rightarrow u[t/x] : B}\ (\text{Cut})$$

$$\frac{\Gamma, x : A, y : B, \Delta \Rightarrow t : C}{\Gamma, y : B, x : A, \Delta \Rightarrow t : C}\ (P)$$

$$\frac{\Delta \Rightarrow t : A \quad \Gamma, x : B \Rightarrow u : C}{\Gamma, \Delta, y : A \to B \Rightarrow u[y(t)/x] : C}\ (\to L) \qquad \frac{\Gamma, x : A \Rightarrow t : B}{\Gamma \Rightarrow \lambda x.t : A \to B}\ (\to R)$$

$$\frac{\Gamma \Rightarrow t : A \quad \Delta \Rightarrow u : B}{\Gamma, \Delta \Rightarrow \langle t, u\rangle : A \circ B}\ (\circ R) \qquad \frac{\Gamma, x : A, y : B \Rightarrow t : C}{\Gamma, z : A \circ B \Rightarrow t[(z)_0/x, (z)_1/y] : C}\ (\circ L)$$

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 $\Lambda(\mathbf{LP})$ 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

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

[2]For 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].

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.

$(E1)$	$(\lambda x^A.t^B)u = t[u/x]$	principal Cut on $A \to B$
$(E2)$	$\lambda x^A.(tx)^B = t$	non-atomic axiom $A \to B$
$(E3)$	$(\langle t^A, u^B\rangle)_0 = t \quad (\langle t, u\rangle)_1 = u$	principal Cut on $A \circ B$
$(E4)$	$\langle (t^{A\circ B})_0, (t^{A\circ B})_1\rangle\rangle = t$	non-atomic axiom $A \circ B$

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:

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

Output:

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

EXAMPLE 3.7. Complex axioms: η-conversion.

$$\dfrac{\dfrac{x : B \Rightarrow x : B \quad y : A \Rightarrow y : A}{v : A/B, x : B \Rightarrow y[v(x)/y] : A}\,[\mathrm{L}/]}{v : A/B \Rightarrow \lambda x.y[v(x)/y] : A/B}\,[\mathrm{R}/] \qquad \rightsquigarrow \qquad \dfrac{}{v : A/B \Rightarrow v : A/B}\,[\mathrm{Ax}]$$

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 7.8). 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.

$$t(A/B) = t(B\backslash A) = t(B) \to t(A), \quad t(A \bullet B) = t(A) \circ t(B) \tag{14}$$

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 ($np\backslash s$) may be mapped to the semantic type $e \to t$ 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.

$$[\text{Ax}]\frac{}{x : A \Rightarrow x : A} \qquad \frac{\Delta \Rightarrow u : A \quad \Gamma[x : A] \Rightarrow t : C}{\Gamma[\Delta] \Rightarrow t[u/x] : C}[\text{Cut}]$$

$$[/\text{R}]\frac{(\Gamma, x : B) \Rightarrow t : A}{\Gamma \Rightarrow \lambda x.t : A/B} \qquad \frac{\Delta \Rightarrow t : B \qquad \Gamma[x : A] \Rightarrow u : C}{\Gamma[(y : A/B, \Delta)] \Rightarrow u[y(t))/x] : C}[/\text{L}]$$

$$[\backslash\text{R}]\frac{(x : B, \Gamma) \Rightarrow t : A}{\Gamma \Rightarrow \lambda x.t : B\backslash A} \qquad \frac{\Delta \Rightarrow t : B \qquad \Gamma[x : A] \Rightarrow u : C}{\Gamma[(\Delta, y : B\backslash A)] \Rightarrow u[y(t)/x] : C}[\backslash\text{L}]$$

$$[\bullet\text{L}]\frac{\Gamma[(x : A, y : B)] \Rightarrow t : C}{\Gamma[z : A \bullet B] \Rightarrow t[(z)_0/x, (z)_1/y] : C} \qquad \frac{\Gamma \Rightarrow t : A \qquad \Delta \Rightarrow u : B}{(\Gamma, \Delta) \Rightarrow \langle t, u \rangle : A \bullet B}[\bullet\text{R}]$$

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, $D_{A\backslash B}$ (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.

$$x : A \vdash x : A$$

$$[/\mathrm{I}]\frac{(\Gamma, x : B) \vdash t : A}{\Gamma \vdash \lambda x.t : A/B} \qquad \frac{\Gamma \vdash t : A/B \qquad \Delta \vdash u : B}{(\Gamma, \Delta) \vdash t(u) : A}[/\mathrm{E}]$$

$$[\backslash\mathrm{I}]\frac{(x : B, \Gamma) \vdash t : A}{\Gamma \vdash \lambda x.t : B\backslash A} \qquad \frac{\Gamma \vdash u : B \qquad \Delta \vdash t : B\backslash A}{(\Gamma, \Delta) \vdash t(u) : A}[\backslash\mathrm{E}]$$

$$[\bullet\mathrm{I}]\frac{\Gamma \vdash t : A \qquad \Delta \vdash u : B}{(\Gamma, \Delta) \vdash \langle t, u\rangle : A \bullet B} \qquad \frac{\Delta \vdash u : A \bullet B \qquad \Gamma[(x : A, y : B)] \vdash t : C}{\Gamma[\Delta] \vdash t[(u)_0/x, (u)_1/y] : C}[\bullet\mathrm{E}]$$

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**.

$$\dfrac{\dfrac{\textbf{needs}}{\text{needs} : (np\backslash s)/((s/np)\backslash s)} \quad \dfrac{\dfrac{\overline{x_1 : s/np} \quad \overline{x_0 : np}}{x_1(x_0) : s}\,/\text{E}}{\lambda x_1.x_1(x_0) : (s/np)\backslash s}\,\backslash\text{I}}{\dfrac{\text{needs}(\lambda x_1.x_1(x_0)) : np\backslash s}{\lambda x_0.\text{needs}(\lambda x_1.x_1(x_0)) : (np\backslash s)/np}\,/\text{I}}\,/\text{E}$$

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 **L** proofs of the theorem $(np\backslash s)/np \Rightarrow ((s/(np\backslash s))\backslash s)/((s/np)\backslash s)$, 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 (†) and object wide scope (‡) reading, respectively.

(†)
$$\dfrac{\dfrac{\overline{x_3 : s/(np\backslash s)} \quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\textbf{tv}}{\textbf{tv} : (np\backslash s)/np} \quad \overline{x_1 : np}}{\textbf{tv}(x_1) : np\backslash s}\,/\text{E} \quad \overline{x_0 : np}}{\textbf{tv}(x_1)(x_0) : s}\,\backslash\text{E}}{\lambda x_1.\textbf{tv}(x_1)(x_0) : s/np}\,/\text{I} \quad \overline{x_2 : (s/np)\backslash s}}{x_2(\lambda x_1.\textbf{tv}(x_1)(x_0)) : s}\,\backslash\text{E}}{\lambda x_0.x_2(\lambda x_1.\textbf{tv}(x_1)(x_0)) : np\backslash s}\,\backslash\text{I}}{x_3(\lambda x_0.x_2(\lambda x_1.\textbf{tv}(x_1)(x_0))) : s}\,/\text{E}}{\lambda x_3.x_3(\lambda x_0.x_2(\lambda x_1.\textbf{tv}(x_1)(x_0))) : (s/(np\backslash s))\backslash s}\,\backslash\text{I}}{\lambda x_2\lambda x_3.x_3(\lambda x_0.x_2(\lambda x_1.\textbf{tv}(x_1)(x_0))) : ((s/(np\backslash s))\backslash s)/((s/np)\backslash s)}\,/\text{I}$$

(‡)
$$\dfrac{\dfrac{\dfrac{\overline{x_2 : s/(np\backslash s)} \quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\textbf{tv}}{\textbf{tv} : (np\backslash s)/np} \quad \overline{x_1 : np}}{\textbf{tv}(x_1) : np\backslash s}\,/\text{E} \quad \overline{x_0 : np}}{\textbf{tv}(x_1)(x_0) : s}\,\backslash\text{E}}{\lambda x_0.\textbf{tv}(x_1)(x_0) : np\backslash s}\,\backslash\text{I}}{x_2(\lambda x_0.\textbf{tv}(x_1)(x_0)) : s}\,/\text{E}}{\lambda x_1.x_2(\lambda x_0.\textbf{tv}(x_1)(x_0)) : s/np}\,/\text{I} \quad \overline{x_3 : (s/np)\backslash s}}{x_3(\lambda x_1.x_2(\lambda x_0.\textbf{tv}(x_1)(x_0))) : s}\,\backslash\text{E}}{\lambda x_2.x_3(\lambda x_1.x_2(\lambda x_0.\textbf{tv}(x_1)(x_0))) : (s/(np\backslash s))\backslash s}\,\backslash\text{I}}{\lambda x_3\lambda x_2.x_3(\lambda x_1.x_2(\lambda x_0.\textbf{tv}(x_1)(x_0))) : ((s/(np\backslash s))\backslash s)/((s/np)\backslash s)}\,/\text{I}$$

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.)

$$\text{himself}_{((np\backslash s)/np)\backslash(np\backslash s)} := \lambda x \lambda y . x(y)(y) \qquad \text{everyone}_{s/(np\backslash s)} := \lambda x \forall y (\text{person}(y) \Rightarrow x(y))$$

$$\dfrac{\dfrac{\textbf{loves}}{\text{loves} : (np\backslash s)/np} \quad \dfrac{}{x_2 : np}}{\text{loves}(x_2) : np\backslash s}\ /E$$

$$\dfrac{\dfrac{}{x_1 : np} \quad \text{loves}(x_2) : np\backslash s}{\text{loves}(x_2)(x_1) : s}\ \backslash E$$

$$\dfrac{\dfrac{\text{loves}(x_2)(x_1) : s}{\lambda x_1.\text{loves}(x_2)(x_1) : np\backslash s}\ \backslash I}{\lambda x_2 \lambda x_1.\text{loves}(x_2)(x_1) : (np\backslash s)/np}\ /I$$

$$\dfrac{\lambda x_2 \lambda x_1.\text{loves}(x_2)(x_1) : (np\backslash s)/np \quad \dfrac{\textbf{himself}}{\text{himself} : ((np\backslash s)/np)\backslash(np\backslash s)}}{\text{himself}(\lambda x_2 \lambda x_1.\text{loves}(x_2)(x_1)) : np\backslash s}\ \backslash E$$

$$\dfrac{\dfrac{}{x_0 : np} \quad \text{himself}(\lambda x_2 \lambda x_1.\text{loves}(x_2)(x_1)) : np\backslash s}{\text{himself}(\lambda x_2 \lambda x_1.\text{loves}(x_2)(x_1))(x_0) : s}\ \backslash E$$

$$\dfrac{\dfrac{\textbf{everyone}}{\text{everyone} : s/(np\backslash s)} \quad \dfrac{\text{himself}(\lambda x_2 \lambda x_1.\text{loves}(x_2)(x_1))(x_0) : s}{\lambda x_0.\text{himself}(\lambda x_2 \lambda x_1.\text{loves}(x_2)(x_1))(x_0) : np\backslash s}\ \backslash I}{\text{everyone}(\lambda x_0.\text{himself}(\lambda x_2 \lambda x_1.\text{loves}(x_2)(x_1))(x_0)) : s}\ /E$$

$$\text{everyone}(\lambda x_0.\text{himself}(\lambda x_2 \lambda x_1.\text{loves}(x_2)(x_1))(x_0)) \rightarrow_\beta \forall y.\text{person}(y) \Rightarrow \text{love}(y)(y)$$

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: $t(s) = t(txt) = s \to s \to t, t(np) = d, t(n) = d \to (s \to (s \to t))$. Lexical recipes for a tiny corner of the DRT lexicon are given in (15).

a^n	$(s/(np\backslash s))/n$	$\lambda P \lambda Q.[u_n \mid]; P(u_n); Q(u_n)$
man,woman	n	$\lambda v.[\mid \textsc{man}\ v], \lambda v.[\mid \textsc{woman}\ v]$
loves, hates	$(np\backslash s)/np$	$\lambda v \lambda w.[\mid v\ \textsc{loves}\ w], \lambda v \lambda w.[\mid v\ \textsc{hates}\ w]$
$\mathrm{he}^n, \mathrm{she}^n$	$s/(np\backslash s)$	$\lambda P.P(u_n)$
$\mathrm{him}^n, \mathrm{her}^n$	$(s/np)\backslash s$	$\lambda P.P(u_n)$
.	$s\backslash(txt/s), txt\backslash(txt/s)$	$\lambda p \lambda q.p; q$

(15)

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:

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

(a) A^1man loves a^2 woman. She_2 hates him_1
(b) $\cdot(\mathrm{a}(\mathrm{man})(\lambda x.\mathrm{a}(\mathrm{woman})(\lambda y.\mathrm{loves}(y)(x))))(\mathrm{she}(\lambda v.\mathrm{him}(\lambda w.\mathrm{hates}(w)(v))))$
(c) $[u_1, u_2 \mid \textsc{man}\ u_1, \textsc{woman}\ u_2, u_1\ \textsc{loves}\ u_2, u_2\ \textsc{hates}\ u_1]$

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 $(e \to t) \to t$, 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 $(e \to t) \to t$ 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: $s/(np\backslash s)$ and $(s/np)\backslash s$. 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 $(s/np)\backslash s$ to 'someone', both the (a) and (a') readings are **L**-derivable. Given a 'subject' assignment $s/(np\backslash s)$ the (b) reading is not derivable: in **L** one only derives the narrow scope reading (b').

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

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 §5.2) 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 $\vec{A} \to B$ abbreviates $A_1 \to \ldots A_n \to B$, and similiarly for $\vec{x}$).

$$
\begin{array}{lrcl}
(AR) & \vec{A} \to B \to \vec{C} \to D & \to & \vec{A} \to ((B \to D) \to D) \to \vec{C} \to D \\
 & t & \to & \lambda \vec{x}_{\vec{A}} \lambda w_{(B \to D) \to D} \lambda \vec{y}_{\vec{C}}.w(\lambda z_B.t(\vec{x})(z)(\vec{y})
\end{array} \tag{17}
$$

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 $A \to B$ 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.

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

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 **(N)L**) 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 $t(q(A, B, C)) = (t(A) \to t(B)) \to t(C)$.

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

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

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

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

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* binderΓ 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 constantsΓ 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.

$$\mathcal{F} ::= \mathcal{A} \mid \mathcal{F}/_i\mathcal{F} \mid \mathcal{F} \bullet_i \mathcal{F} \mid \mathcal{F}\backslash_i\mathcal{F} \tag{19}$$

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 $\langle W, \{R_i^3\}_{i\in I}\rangle$.

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

$$A \to C/_i B \quad \textit{iff} \quad A \bullet_i B \to C \quad \textit{iff} \quad B \to A\backslash_i C \tag{20}$$

In sequent presentation, each residuated family of multiplicatives $\{/_i, \bullet_i, \backslash_i\}$ 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 $\mathcal{S} ::= \mathcal{F} \mid (\mathcal{S},\mathcal{S})^i$.

$$[\mathrm{R}/_i]\frac{(\Gamma, B)^i \Rightarrow A}{\Gamma \Rightarrow A/_i B} \qquad \frac{\Gamma \Rightarrow B \qquad \Delta[A] \Rightarrow C}{\Delta[(A/_i B, \Gamma)^i] \Rightarrow C}[\mathrm{L}/_i]$$

$$[\mathrm{R}\backslash_i]\frac{(B, \Gamma)^i \Rightarrow A}{\Gamma \Rightarrow B\backslash_i A} \qquad \frac{\Gamma \Rightarrow B \qquad \Delta[A] \Rightarrow C}{\Delta[(\Gamma, B\backslash_i A)^i] \Rightarrow C}[\mathrm{L}\backslash_i]$$

$$[\mathrm{L}\bullet_i]\frac{\Gamma[(A, B)^i] \Rightarrow C}{\Gamma[A \bullet_i B] \Rightarrow C} \qquad \frac{\Gamma \Rightarrow A \qquad \Delta \Rightarrow B}{(\Gamma, \Delta)^i \Rightarrow A \bullet_i B}[\mathrm{R}\bullet_i]$$

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 $\bullet_c$: $(\forall x, y, z \in W)\ R_c xyz \Rightarrow R_c xzy$. (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.)

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

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 $(\forall xyz \in W)\ R_i xyz \Rightarrow R_j xyz$, 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 & Oehrle 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 commutativity 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.

$$\begin{array}{lll} MP: & A \bullet_i (B \bullet_j C) \longleftrightarrow B \bullet_j (A \bullet_i C) & \exists t(R_i uxt \ \& \ R_j tyz) \Leftrightarrow \exists t'(R_j uyt' \ \& \ R_i t'xz) \\ MA: & A \bullet_i (B \bullet_j C) \longleftrightarrow (A \bullet_i B) \bullet_j C & \exists t(R_i uxt \ \& \ R_j tyz) \Leftrightarrow \exists t'(R_j ut'z \ \& \ R_i t'xy) \end{array}$$

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

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 $(vp/_h iv, np\backslash_r iv)^h \Rightarrow np\backslash_r vp$ with Ex 2.22.)

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

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 $(s/_w np)\backslash_w s$. Associativity here is essential for obtaining access to arbitrary infixation points for the wrapping expression.

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

Example 4.5. Multimodal deconstruction of $q(A, B, C)$ as $(B/_w A)\backslash_w C$. On the left the [q L] rule of Def 3.14. On the right the 'partial execution' compilation in terms of interaction principle (22).

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

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.

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

$$(R_i txy\ \&\ R_j utz) \Rightarrow \exists t' \exists t'' (R_j t' xz\ \&\ R_j t'' yz\ \&\ R_i ut't'')$$

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].

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

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 $(A\backslash_iB)/_jC$ as $(vp\backslash_ivp)/_jnp$ 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.

$$\dfrac{\dfrac{\dfrac{\&c}{((A/_jC, C)^j, (A\backslash_iB)/_jC, C)^j)^i \Rightarrow B}\, MC}{((A/_jC, (A\backslash_iB)/_jC)^i, C)^j \Rightarrow B}\, /_jR}{(A/_jC, (A\backslash_iB)/_jC)^i \Rightarrow B/_jC}$$

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 $f_\bullet(A_1, \ldots, A_n)$ for an n-ary product and $f^i_\rightarrow(A_1, \ldots, A_n)$ 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].

$$f_\bullet(A_1, \ldots, A_n) \rightarrow B \qquad \textit{iff} \qquad A_i \rightarrow f^i_\rightarrow(A_1, \ldots, A_{i-1}, B, A_{i+1}, \ldots, A_n) \tag{24}$$

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 $\Diamond$, $\Box^{\downarrow}$. Parallel to our treatment of the binary multiplicatives §2, we start from the most discriminating system, i.e. the pure logic of residuation for $\Diamond$, $\Box^{\downarrow}$. 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 $\Diamond$, $\Box^{\downarrow}$.

DEFINITION 4.8. Unary multiplicative connectives: the pure logic of residuation. Interpretation clauses. Residuation laws. Note that the interpretation of $\Diamond$ and $\Box^{\downarrow}$ 'moves' in opposite directions along the R^2 accessibility relation. (The downarrow on the universal operator is there to highlight this fact.)

$$v(\Diamond A) = \{x \mid \exists y(Rxy \ \wedge \ y \in v(A)\}$$
$$v(\Box^{\downarrow} A) = \{x \mid \forall y(Ryx \ \Rightarrow y \in v(A)\}$$

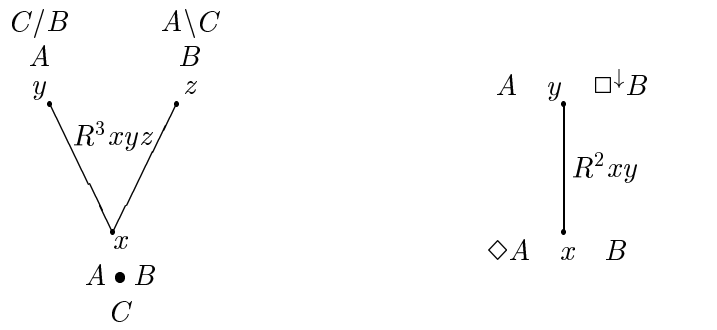
$$\Diamond A \to B \quad \textit{iff} \quad A \to \Box^{\downarrow} B$$

COMPLETENESS. The completeness result of Prop 2.3 for the binary multiplicative language extends unproblematically to the language enriched with $\Diamond, \Box^{\downarrow}$. 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 $\vdash A \to \Diamond B$. The Truth Lemma has to be checked for the new compound formulae $\Diamond A$, $\Box^{\downarrow} A$. The direction that requires a little thinking is dealt with below.

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

($\Box^{\downarrow}$) Assume $A \in v(\Box^{\downarrow} B)$. We have to show $\vdash A \to \Box^{\downarrow} B$. $A \in v(\Box^{\downarrow} B)$ 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 $\vdash \Diamond A \to \Diamond A$. By induction hypothesis we have $\vdash A' \to B$, i.e. $\vdash \Diamond A \to B$. By Residuation this gives $\vdash A \to \Box^{\downarrow} B$.

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 $\Box^{\downarrow}$ 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



$A \to C/B \quad \textit{iff} \quad A \bullet B \to C$
$A \bullet B \to C \quad \textit{iff} \quad B \to A\backslash C$

$\Diamond A \to B \quad \textit{iff} \quad A \to \Box^{\downarrow} B$

Figure 1: Kripke graphs: binary and unary multiplicatives

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 5.6).

In our discussion of the binary vocabulary in §2, we pointed out that one can characterize $/, \bullet, \backslash$ 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 $\Diamond\Box^{\downarrow}$, $\Box^{\downarrow}\Diamond$, and Isotonicity.

$$(\dagger) \qquad \frac{f : \Diamond A \to B}{\mu(f) : A \to \Box^{\downarrow} B} \qquad \frac{g : A \to \Box^{\downarrow} B}{\mu^{-1}(g) : \Diamond A \to B}$$

$$(\ddagger) \qquad \begin{array}{rl} \textbf{unit}_{\Box^{\downarrow}} : & \Diamond\Box^{\downarrow} A \to A \\ \textbf{co-unit}_{\Box^{\downarrow}} : & A \to \Box^{\downarrow}\Diamond A \end{array} \qquad \frac{f : A \to B}{(f)^{\Diamond} : \Diamond A \to \Diamond B} \qquad \frac{f : A \to B}{(f)^{\Box^{\downarrow}} : \Box^{\downarrow} A \to \Box^{\downarrow} B}$$

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 $\Diamond\Box^{\downarrow}$ and $\Box^{\downarrow}\Diamond$ (Term decoration for the right column left to the reader.)

$$\frac{1_{\Box^{\downarrow} A} : \Box^{\downarrow} A \to \Box^{\downarrow} A}{\mu^{-1}(1_{\Box^{\downarrow} A}) : \Diamond\Box^{\downarrow} A \to A} \qquad \frac{\Diamond A \to \Diamond A}{A \to \Box^{\downarrow}\Diamond A} \tag{25}$$

$$\frac{f : A \to B \quad \dfrac{1_{\Diamond B} : \Diamond B \to \Diamond B}{\mu(1_{\Diamond B}) : B \to \Box^{\downarrow}\Diamond B}}{\dfrac{\mu(1_{\Diamond B}) \circ f : A \to \Box^{\downarrow}\Diamond B}{\mu^{-1}(\mu(1_{\Diamond B}) \circ f) : \Diamond A \to \Diamond B}} \qquad \frac{\dfrac{\Box^{\downarrow} A \to \Box^{\downarrow} A}{\Diamond\Box^{\downarrow} A \to A} \quad A \to B}{\dfrac{\Diamond\Box^{\downarrow} A \to B}{\Box^{\downarrow} A \to \Box^{\downarrow} B}} \tag{26}$$

GENTZEN CALCULUS. Following the agenda set out in §2 for the binary connectives, we introduce Gentzen sequent rules for the connectives $\Diamond, \Box^{\downarrow}$. 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 $/, \bullet, \backslash$, and unary $(\cdot)^\diamond$ for the family $\diamond, \Box^\downarrow$.

$$\mathcal{F} ::= \mathcal{A} \mid \mathcal{F}/\mathcal{F} \mid \mathcal{F} \bullet \mathcal{F} \mid \mathcal{F}\backslash\mathcal{F} \mid \diamond\mathcal{F} \mid \Box^\downarrow\mathcal{F} \qquad \mathcal{S} ::= \mathcal{F} \mid (\mathcal{S},\mathcal{S}) \mid (\mathcal{S})^\diamond \tag{27}$$

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

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

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

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 $\diamond, \Box^\downarrow$ does not depend on semantic restrictions on the R^2 composition relation. In addition to the fixed 'logical' part of the $\diamond, \Box^\downarrow$ connectives, we can consider various structural resource management options for the unary family $\diamond, \Box^\downarrow$ 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^2_j for $\langle j\rangle, [j]^\downarrow$ in one logic. The multimodal generalization is completely standard.

DEFINITION 4.11. Unary connectives: resource management options. Structural postulates, frame constraints, Gentzen rules. (For $\Box^\downarrow$ duals of these postulates: replace $\diamond$ by $\Box^\downarrow$ and reverse the arrow.)

$4:$	$\diamond\diamond A \to \diamond A$	$(Rxy \ \&\ Ryz) \Rightarrow Rxz$
$T:$	$A \to \diamond A$	Rxx
$K1:$	$\diamond(A \bullet B) \to \diamond A \bullet B$	$(Rwx \ \&\ Rxyz) \Rightarrow \exists y'(Ry'y \ \&\ Rwy'z)$
$K2:$	$\diamond(A \bullet B) \to A \bullet \diamond B$	$(Rwx \ \&\ Rxyz) \Rightarrow \exists z'(Rz'z \ \&\ Rwyz')$
$K:$	$\diamond(A \bullet B) \to \diamond A \bullet \diamond B$	$(Rwx \ \&\ Rxyz) \Rightarrow \exists y'\exists z'(Ry'y \ \&\ Rz'z \ \&\ Rwy'z')$

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

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

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 $\Box^\downarrow$ 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 $(\cdot)^\diamond$ 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 $(\cdot)^\diamond$ structural punctuation. We write $\Box^\downarrow\Gamma$, $(\Box^\downarrow)^\diamond\Gamma$, $(\Box^\downarrow\Box^\downarrow)^\diamond\Gamma$ for a term Γ of which the (pre)terminal subterms are all of the form $\Box^\downarrow A$, $(\Box^\downarrow A)^\diamond$, $(\Box^\downarrow\Box^\downarrow A)^\diamond$, respectively. The $4(Cut)$ step is a series of replacements (read bottom-up) of terminal $\Box^\downarrow A$ by $\Box^\downarrow\Box^\downarrow A$ via Cuts depending on 4.

$$\frac{\dfrac{\Gamma[A] \Rightarrow B}{\Gamma[(\Box^\downarrow A)^\diamond] \Rightarrow B}\ \Box^\downarrow L}{\Gamma[\Box^\downarrow A] \Rightarrow B}\ T \qquad \rightsquigarrow \qquad \frac{\Gamma[A] \Rightarrow B}{\Gamma[\Box^\downarrow A] \Rightarrow B}\ \Box^\downarrow L(S4)$$

$$\frac{\dfrac{\dfrac{\dfrac{\dfrac{\Box^\downarrow\Gamma \Rightarrow A}{(\Box^\downarrow\Box^\downarrow)^\diamond\Gamma \Rightarrow A}\ \Box^\downarrow L}{(\Box^\downarrow)^\diamond\Gamma \Rightarrow A}\ 4(Cut)}{(\Box^\downarrow\Gamma)^\diamond \Rightarrow A}\ K}{\Box^\downarrow\Gamma \Rightarrow \Box^\downarrow A}\ \Box^\downarrow R \qquad \rightsquigarrow \qquad \frac{\Box^\downarrow\Gamma \Rightarrow A}{\Box^\downarrow\Gamma \Rightarrow \Box^\downarrow A}\ \Box^\downarrow R(S4)$$

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 $\diamond, \Box^\downarrow$ 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 $\diamond, \Box^\downarrow$ 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 $\Box$ 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 §5, 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 $\Diamond, \Box^{\downarrow}$ rather than a single modal operator $\Box$, 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 $\Box$ become valid type transitions in the pure residuation system for $\Diamond, \Box^{\downarrow}$, 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 4 via compilation $(\Box A)^\sharp = \Diamond\Box^{\downarrow}(A)^\sharp$.

$$\begin{array}{lll} T: & \Box A \to A & \rightsquigarrow \quad \Diamond\Box^{\downarrow} A \to A \\ 4: & \Box A \to \Box\Box A & \rightsquigarrow \quad \Diamond\Box^{\downarrow} A \to \Diamond\Box^{\downarrow}\Diamond\Box^{\downarrow} A \end{array}$$

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

$$(P_\Diamond): \Diamond A \bullet B \to B \bullet \Diamond A \qquad (A_\Diamond): (A_1 \bullet A_2) \bullet A_3 \longleftrightarrow A_1 \bullet (A_2 \bullet A_3) (\dagger)$$

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

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 $\Diamond, \Box^{\downarrow}$. Destructors ${}^{\cup}\cdot$ and ${}^{\vee}\cdot$, corresponding to rules of use for $\Diamond$ and $\Box^{\downarrow}$. Constructors ${}^{\cap}\cdot$ and ${}^{\wedge}\cdot$, for rules of proof. Compare Def 3.1 for the binary vocabulary.

$$\mathcal{M}^A ::= \ldots \mid {}^{\cup}(\mathcal{M}^{\Diamond A}) \mid {}^{\vee}(\mathcal{M}^{\Box^{\downarrow} A}) \qquad \mathcal{M}^{\Diamond A} ::= {}^{\cap}(\mathcal{M}^A) \qquad \mathcal{M}^{\Box^{\downarrow} A} ::= {}^{\wedge}(\mathcal{M}^A)$$

DEFINITION 4.16. Term assignment. The $\Diamond, \Box^{\downarrow}$ cases.

$$\frac{\Gamma \Rightarrow t : A}{(\Gamma)^\Diamond \Rightarrow {}^{\cap}t : \Diamond A}\ \Diamond R \qquad \frac{\Gamma[(y : A)^\Diamond] \Rightarrow t : B}{\Gamma[x : \Diamond A] \Rightarrow t[{}^{\cup}x/y] : B}\ \Diamond L$$

$$\frac{(\Gamma)^\Diamond \Rightarrow t : A}{\Gamma \Rightarrow {}^{\wedge}t : \Box^{\downarrow} A}\ \Box^{\downarrow} R \qquad \frac{\Gamma[y : A] \Rightarrow t : B}{\Gamma[(x : \Box^{\downarrow} A)^\Diamond] \Rightarrow t[{}^{\vee}x/y] : B}\ \Box^{\downarrow} L$$

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

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

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.

$$\frac{\Box\Gamma \Rightarrow t : A}{\Box\Gamma \Rightarrow \hat{}\, t : \Box A}\ \Box R \qquad \frac{\Gamma, x : A, \Gamma' \Rightarrow t : B}{\Gamma, y : \Box A, \Gamma' \Rightarrow t[\check{}\, y/x] : B}\ \Box L \tag{28}$$

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.

$$\begin{array}{rl} & \text{(the book) that Kazimierz wrote} \\ & \text{(the book) that Kazimierz wrote yesterday} \\ \mathbf{L} \vdash & r/(s/np), np, (np\backslash s)/np \Rightarrow r \\ \mathbf{L} \nvdash & r/(s/np), np, (np\backslash s)/np, s\backslash s \Rightarrow r \\ \mathbf{NL} \nvdash & (r/(s/np), (np, (np\backslash s)/np)) \Rightarrow r \end{array} \tag{29}$$

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 $\Diamond\Box^{\downarrow}$ with the derivation of controlled rebracketing within **NL**.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\&c}{(np,(tv,np))\Rightarrow s}}{(np,(tv,(\Box^{\downarrow}_a np)^{\diamond}))\Rightarrow s}\,\Box^{\downarrow}L}{((np,tv),(\Box^{\downarrow}_a np)^{\diamond})\Rightarrow s}\,A_{\diamond}}{((np,tv),\Diamond_a\Box^{\downarrow}_a np)\Rightarrow s}\,\Diamond L}{(np,tv)\Rightarrow s/\Diamond_a\Box^{\downarrow}_a np}\,/R \tag{30}$$

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 overgeneration 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
	L $\vdash$	$r/(s/np), np, tv, (X\backslash X)/X, np, tv \Rightarrow r \quad (X = s/np)$
	L$\Diamond$ $\vdash$	$r/(s/np), (np, tv, (X\backslash\Box^{\downarrow}X)/X, np, tv)^{\diamond} \Rightarrow r$
b.		*(the logician) whom Gottlob admired Jim and Kazimierz detested
	L $\vdash$	$r/(s/np), np, tv, np, (X\backslash X)/X, np, tv \Rightarrow r \quad (X = s)$
	L$\Diamond$ $\nvdash$	$r/(s/np), (np, tv, np, (X\backslash\Box^{\downarrow}X)/X, np, tv)^{\diamond} \Rightarrow r$

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': $(X\backslash\Box^{\downarrow}X)/X$. (We recast the analysis in terms of the pure residuation logic for $\Diamond, \Box^{\downarrow}$.) 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 $\Box^{\downarrow}X$. To eliminate the $\Box^{\downarrow}$ 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 $\Diamond\Box^{\downarrow}X \to X$. 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 $\Diamond, \Box^{\downarrow}$ 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 $\Diamond, \Box^{\downarrow}$ decoration.

	RESOURCE	GOAL
$\Diamond$	domain-erasure	domain-projection
$\Box^{\downarrow}$	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 $\mathbf{L}$ type-assignment $((np\backslash s)/np)\backslash(np\backslash s)$ for 'himself' with meaning recipe $\lambda x\lambda y.x(y)(y)$. Within $\mathbf{L}$, (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 $((np\backslash s)/np)$ and a complex expression of that type which has been composed across clausal boundaries.

a.	David admires himself	$\mathbf{L}\vdash$	$(np\backslash s)/np \Rightarrow (np\backslash s)/np$
b.	David cares for himself	$\mathbf{L}\vdash$	$(np\backslash s)/pp, pp/np \Rightarrow (np\backslash s)/np$
c.	*David thinks Emmy admires himself	$\mathbf{L}\vdash$	$(np\backslash s)/s, np, (np\backslash s)/np \Rightarrow (np\backslash s)/np$

(32)

Within $\mathbf{L}+\Box$, appropriate modalization provides lexical control to make (a) and (b) derivable while ruling out (c). In moving from $\mathbf{L}$ to $\mathbf{L}+\Box$ lexical type assignments, one prefixes the original $\mathbf{L}$ 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.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\text{FAIL}}{\Box np, \Box((np\backslash s)/np), np \Rightarrow \Box s}\quad \dfrac{\&c}{np, np\backslash s \Rightarrow s}\,/L}{np, (np\backslash s)/\Box s, \Box np, \Box((np\backslash s)/np), np \Rightarrow s}\,\Box L}{np, \Box((np\backslash s)/\Box s), \Box np, \Box((np\backslash s)/np), np \Rightarrow s}\,/R,\backslash R \quad \dfrac{\dfrac{\&c}{np, np\backslash s \Rightarrow s}\,\Box L}{\Box np, np\backslash s \Rightarrow s}\,\backslash L}{\Box np, \Box((np\backslash s)/\Box s), \Box np, \Box((np\backslash s)/np), ((np\backslash s)/np)\backslash(np\backslash s) \Rightarrow s}\,\Box L}{\Box np, \Box((np\backslash s)/\Box s), \Box np, \Box((np\backslash s)/np), \Box((np\backslash s)/np)\backslash(np\backslash s) \Rightarrow s}\,\Box L}{\Box np, \Box((np\backslash s)/\Box s), \Box np, \Box((np\backslash s)/np), \Box((np\backslash s)/np)\backslash(np\backslash s) \Rightarrow \Box s}\,\Box R$$

*David thinks Emmy loves himself

A more elaborate account of syntactic island constraints is offered in [Hepple 90, Hepple 92] in terms of a *polymodal* system with domain modalities $\{\Box_j\}_{j\in J}$. The domain modalities have an order defined on them, which allows for the characterization of syntactic boundaries of different strength. Island constraints are lexically controlled through the interplay of type-assignment to complement taking functors and 'extractable' elements. Take a relative pronoun with type $\Box((n\backslash n)/(s/\Box_i np))$ and a verb subcategorizing for a clausal complement,

$\Box((np\backslash s)/\Box_j s)$. The relative pronoun will be extractable from the $\Box_j s$ embedded clause provided $\Box_i \preceq \Box_j$.

We have presented the analysis of locality domains in terms of the original $S4$ decoration of [Hepple 90]. Decomposing the $S4$ account into its structural components, we see that the checking of uniform antecedent $\Box$ marking is taken care of by the K distributivity principle of Def 4.11. In fact, with a slightly adapted modalization strategy which decorates the assignment to 'think' as $\Box^{\downarrow}(np\backslash s)/\Box^{\downarrow}s$, one can recast the above analysis in terms of K and the $\Diamond, \Box^{\downarrow}$ residuation logic, as the reader can check. The same combination of RES$\Diamond, \Box^{\downarrow} + K$ lies at the basis of an analysis of French clitic pronouns in [Kraak 95], and of the type-logical account of Linear Precedence constraints in [Versmissen 96].

4.2.3 Resource control: faithful embeddings

In §4.2.2 we have presented analyses of a number of linguistic phenomena which rely on modally decorated type-assignments to obtain structural relaxation, or to impose structural constraints. These applications suggest a more fundamental logical question: Can one provide a *general* theory of resource control in terms of the unary vocabularyΓ The embedding theorems of [Kurtonina & Moortgat 95] answer this question in the affirmative: they show that the $\Diamond, \Box^{\downarrow}$ connectives provide a theory of systematic communication between the type logics of Fig 2. Below, we discuss the strategies for modal decoration realizing the embeddings, and reflect on general logical and linguistic aspects of this approach.

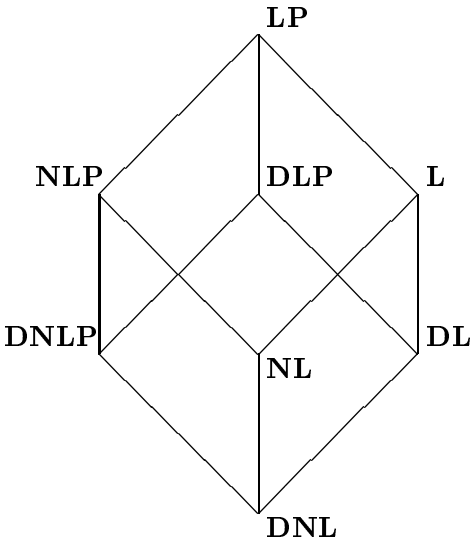


Figure 2: Resource-sensitive logics: precedence, dominance, dependency

Figure 2 displays the resource logics one obtains in terms of the structural parameters of precedence (word-order), dominance (constituent structure) and dependency. The systems occupying the upper plane of Figure 2 were the subject of §2. As we have seen in our discussion of Def 4.2 each of these systems has a dependency variant, where the product is split up into a left-headed $\bullet_l$ and a right-headed $\bullet_r$ version.

Consider a pair of logics $\mathcal{L}_0, \mathcal{L}_1$ where $\mathcal{L}_0$ is a 'southern' neighbour of $\mathcal{L}_1$. Let us write

$\mathcal{L}\Diamond$ for the system $\mathcal{L}$ extended with the unary operators $\Diamond, \Box^{\downarrow}$ with their minimal residuation logic. For the 12 edges of the cube of Fig 2, one can define *embedding translations* $(\cdot)^{\flat} : \mathcal{F}(\mathcal{L}_0) \mapsto \mathcal{F}(\mathcal{L}_1\Diamond)$ which impose the structural discrimination of $\mathcal{L}_0$ in $\mathcal{L}_1$ with its more liberal resource management, and $(\cdot)^{\sharp} : \mathcal{F}(\mathcal{L}_1) \mapsto \mathcal{F}(\mathcal{L}_0\Diamond)$ which license relaxation of structure sensitivity in $\mathcal{L}_0$ in such a way that one fully recovers the flexibility of the the coarser $\mathcal{L}_1$. The embedding translations decorate critical subformulae in the target logic with the operators $\Diamond, \Box^{\downarrow}$. The translations are defined on the product $\bullet$ of the source logic: their action on the implicational formulas is fully determined by the residuation laws. For the $\cdot^{\flat}$ type of embedding, the modal decoration has the effect of blocking 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 $\cdot^{\flat}$ 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 $\cdot^{\flat} : \mathcal{F}(\mathbf{NL}) \mapsto \mathcal{F}(\mathbf{L}\Diamond)$, and $\cdot^{\sharp} : \mathcal{F}(\mathbf{L}) \mapsto \mathcal{F}(\mathbf{NL}\Diamond)$.

$$\begin{array}{cc} p^{\flat} = p & p^{\sharp} = p \\ (A \bullet_0 B)^{\flat} = \Diamond(A^{\flat} \bullet_1 B^{\flat}) & (A \bullet_1 B)^{\sharp} = \Diamond(A^{\sharp} \bullet_0 B^{\sharp}) \\ (A/_0 B)^{\flat} = \Box^{\downarrow}A^{\flat}/_1 B^{\flat} & (A/_1 B)^{\sharp} = \Box^{\downarrow}A^{\sharp}/_0 B^{\sharp} \\ (B\backslash_0 A)^{\flat} = B^{\flat}\backslash_1 \Box^{\downarrow}A^{\flat} & (B\backslash_1 A)^{\sharp} = B^{\sharp}\backslash_0 \Box^{\downarrow}A^{\sharp} \end{array}$$

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

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

$$\mathbf{NL} \vdash A \to B \quad \text{iff} \quad \mathbf{L}\Diamond \vdash A^{\flat} \to B^{\flat}$$

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 $(\cdot)^{\sharp}$, $(A_{\Diamond})$.

$$\mathcal{L}_1 : \quad A \bullet_1 (B \bullet_1 C) \longleftrightarrow (A \bullet_1 B) \bullet_1 C \quad (A)$$

$$\mathcal{L}_0 : \quad \Diamond(A \bullet_0 \Diamond(B \bullet_0 C)) \longleftrightarrow \Diamond(\Diamond(A \bullet_0 B) \bullet_0 C) \quad (A_{\Diamond})$$

PROPOSITION 4.23. Dominance structure: licensing relaxation ([Kurtonina & Moortgat 95]).

$$\mathbf{L} \vdash A \to B \quad \text{iff} \quad \mathbf{NL}\Diamond + A_{\Diamond} \vdash A^{\sharp} \to B^{\sharp}$$

The derivations of Ex 4.24 illustrate the complementary strategies with the Geach rule, the characteristic theorem which differentiates **L** from **NL**. On the left, we try to derive the $\cdot^\flat$ translation of the Geach rule in $\mathbf{L}\diamond$. The resource management regime is associative — still the derivation fails because of the structural $(\cdot)^\diamond$ decoration which makes the C resource inaccessible for the functor $\Box^\downarrow B/_1 C$. On the right one finds a successful derivation of the $\cdot^\sharp$ translation in $\mathbf{NL}\diamond$. Although the resource management regime in this case does not allow free rebracketing, the $\diamond$ decoration gives access to the modal version of the structural rule.

EXAMPLE 4.24. Imposing structural control versus relaxing structure sensitivity.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\text{FAIL}}{((((\Box^\downarrow A/_1 B, \Box^\downarrow B/_1 C)^1)^\circ, C)^1)^\circ \Rightarrow A}\,\Box^\downarrow R}{(((\Box^\downarrow A/_1 B, \Box^\downarrow B/_1 C)^1)^\circ, C)^1 \Rightarrow \Box^\downarrow A}\,/_1 R}{((\Box^\downarrow A/_1 B, \Box^\downarrow B/_1 C)^1)^\circ \Rightarrow \Box^\downarrow A/_1 C}\,\Box^\downarrow R}{(\Box^\downarrow A/_1 B, \Box^\downarrow B/_1 C)^1 \Rightarrow \Box^\downarrow(\Box^\downarrow A/_1 C)}\,/_1 R}{\Box^\downarrow A/_1 B \Rightarrow \Box^\downarrow(\Box^\downarrow A/_1 C)/_1(\Box^\downarrow B/_1 C)}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{C \Rightarrow C \quad \dfrac{B \Rightarrow B}{(\Box^\downarrow B)^\circ \Rightarrow B}\,\Box^\downarrow L}{((\Box^\downarrow B/_0 C, C)^0)^\circ \Rightarrow B}\,/_0 L \quad \dfrac{A \Rightarrow A}{(\Box^\downarrow A)^\circ \Rightarrow A}\,\Box^\downarrow L}{((\Box^\downarrow A/_0 B, ((\Box^\downarrow B/_0 C, C)^0)^\circ)^0)^\circ \Rightarrow A}\,/_0 L}{((((\Box^\downarrow A/_0 B, \Box^\downarrow B/_0 C)^0)^\circ, C)^0)^\circ \Rightarrow A}\,A_\diamond}{(((\Box^\downarrow A/_0 B, \Box^\downarrow B/_0 C)^0)^\circ, C)^0 \Rightarrow \Box^\downarrow A}\,\Box^\downarrow R}{((\Box^\downarrow A/_0 B, \Box^\downarrow B/_0 C)^0)^\circ \Rightarrow \Box^\downarrow A/_0 C}\,/_0 R}{(\Box^\downarrow A/_0 B, \Box^\downarrow B/_0 C)^0 \Rightarrow \Box^\downarrow(\Box^\downarrow A/_0 C)}\,\Box^\downarrow R}{\Box^\downarrow A/_0 B \Rightarrow \Box^\downarrow(\Box^\downarrow A/_0 C)/_0(\Box^\downarrow B/_0 C)}\,/_0 R}$$

$$\mathbf{L}\diamond \not\vdash (A/_0 B)^\flat \Rightarrow ((A/_0 C)/_0(B/_0 C))^\flat \qquad \mathbf{NL}\diamond + (A_\diamond) \vdash (A/_1 B)^\sharp \Rightarrow ((A/_1 C)/_1(B/_1 C))^\sharp$$

DISCUSSION. With respect to the theme of resource control it is instructive to contrast Linear Logic with the grammar logics discussed here. The theory of communication presented above uses the standard logical technique of embeddings. In Linear Logic, the unary 'exponentials' are designed to recover the expressivity of the structural rules of Contraction and Weakening in a controlled way. The modalities that achieve the desired embedding are governed by an $S4$-like regime. The 'sublinear' grammar logics exhibit a higher degree of structural organization. These more discriminating logics suggest more delicate instruments for obtaining structural control: as we have seen, the pure residuation logic for $\diamond, \Box^\downarrow$ does not depend on specific assumptions about the grammatical composition relation R^2, but it is expressive enough to obtain full control over grammatical resource management.[3] A second difference with the Linear Logic approach is the *bi-directionality* of the proposed communication: from the grammatical point of view, imposing structural constraints and licensing structural relaxation are equally significant forms of resource control.

On the level of actual grammar development, the embedding results provide a solution to the problem of 'mode proliferation' inherent in the multimodal approach of §4.1. The multimodal style of grammatical reasoning relies heavily on a (potentially unmanageable) inventory of primitive composition modes $\bullet_i$. The control operators $\diamond, \Box^\downarrow$ make it possible to reanalyse the various $\bullet_i$ as *defined* connectives, in terms of a familiar $\bullet$ and modal decoration. The dependency connectives $\bullet_l$,$\bullet_r$, for example, can be introduced as synthetic operators with definitions $(\diamond -) \bullet -$, and $- \bullet (\diamond -)$, respectively, with $\diamond$ marking the head component. This perspective suggests a global division of labour between 'syntax' and 'semantics', with **LP** playing the role of the default semantic composition language, and the pure residuation

[3] It is interesting to note that for reasons different from ours, and for different types of models, a number of proposals in the field of Linear Logic have argued for a decomposition of the exponentials into more elementary operators (cf. [Bucalo 94, Girard 95b]).

logic **NL** the default language of structural composition. The intermediate territory can be navigated by means of the modal control operators.

5 Reasoning about multiple type assignments

In the previous sections, we have discussed the multiplicative vocabulary, and its interpretation in terms of structural composition of grammatical resources. If we want to carry out the lexicalist program and account for the composition of form and meaning in purely deductive terms, it will often be necessary to associate grammatical resources with *multiple* types — types which are not related in terms of derivability. It is of vital importance, then, that the grammar logic also provides facilities for reasoning *about* such multiple type assignments. In the present section, we discuss two extensions of the type-logical vocabulary that serve this purpose. In §5.1, we introduce Boolean and additive type constructors for conjunctive/disjunctive types of reasoning. In §5.2, the type language is extended with first and second order quantifiers. These constructors express generalizations accross types in terms of 'dependent' types and type schemata.

5.1 Additive and boolean operations

Let us extend the language of type formulae with logical constants $\sqcap, \sqcup$. A natural interpretation for formulas $A \sqcap B, A \sqcup B$ can be given in terms of the intersection and union of the interpretations of the subformulas A and B.

$$\begin{array}{l} v(A \sqcap B) = \{x \mid x \in v(A) \wedge x \in v(B)\} = v(A) \cap v(B) \\ v(A \sqcup B) = \{x \mid x \in v(A) \vee x \in v(B)\} = v(A) \cup v(B) \end{array} \tag{33}$$

In an axiomatic presentation of the type logic, one introduces appropriate axiom schemata and rules of inference for the new type constructors. It is shown in [Kurtonina 95] that with respect to the frame semantics for the multiplicatives, completeness is maintained when the $\sqcap, \sqcup$ connectives are added. Instead of the simple formula-based canonical model construction, the completeness proof now employs a hierarchy of filter constructions.

Definition 5.1. Lattice operations. Axioms, rules of inference.

$$\pi^1_{A,B} : A \sqcap B \to A \qquad \pi^2_{A,B} : A \sqcap B \to B \qquad \frac{f : C \to A \quad g : C \to B}{\langle f, g \rangle : C \to A \sqcap B}$$

$$\kappa^1_{A,B} : A \to A \sqcup B \qquad \kappa^2_{A,B} : B \to A \sqcup B \qquad \frac{f : A \to C \quad g : B \to C}{[f, g] : A \sqcup B \to C}$$

As a first approximation of a Gentzen formulation, one could follow the usual route, replacing on the lefthand side of the derivability arrow formula variables by structure variables, and contextually embedding the $\pi^i, [f, g]$ schemata in such a way that the inference rules have the proper format for Cut Elimination. Cf. [Došen 89, Lambek 93a]. One obtains the sequent rules for the *additives* of Linear Logic.

DEFINITION 5.2. Gentzen rules: additives.

$$\frac{\Gamma[A] \Rightarrow C}{\Gamma[A \sqcap B] \Rightarrow C}\ \sqcap L \qquad \frac{\Gamma[B] \Rightarrow C}{\Gamma[A \sqcap B] \Rightarrow C}\ \sqcap L \qquad \frac{\Gamma \Rightarrow A \quad \Gamma \Rightarrow B}{\Gamma \Rightarrow A \sqcap B}\ \sqcap R$$

$$\frac{\Gamma \Rightarrow A}{\Gamma \Rightarrow A \sqcup B}\ \sqcup R \qquad \frac{\Gamma \Rightarrow B}{\Gamma \Rightarrow A \sqcup B}\ \sqcup R \qquad \frac{\Gamma[A] \Rightarrow C \quad \Gamma[B] \Rightarrow C}{\Gamma[A \sqcup B] \Rightarrow C}\ \sqcup L$$

The Curry-Howard interpretation for the extended vocabulary uses pairing and projection for term assignment to the $\sqcap$ connective[4], and a disjoint sum and case construction operation for the interpretation of $\sqcup$, cf. [Troelstra 92]. (34) extends the syntax of lambda terms with clauses for the $\sqcup$ connective.

$$\mathcal{T}_A ::= \ldots \mid \mathrm{E}^{\sqcup}_{\mathcal{V}_B,\mathcal{V}_C}(\mathcal{T}_{B\sqcup C}, \mathcal{T}_A, \mathcal{T}_A) \quad \mathcal{T}_{A\sqcup B} ::= \kappa_1 \mathcal{T}_A \quad \mathcal{T}_{B\sqcup A} ::= \kappa_2 \mathcal{T}_A \tag{34}$$

The relevant term equations for disjoint sum and case construction are given below. In a term $E^{\sqcup}_{x,y}(s,t,u)$ the subscript on the constructor $E^{\sqcup}_{x,y}$ indicates that the variables x and y are bound in t and u, respectively.

$$\mathrm{E}^{\sqcup}_{x,y}(\kappa_1 s, t, u) = t[s/x] \qquad \mathrm{E}^{\sqcup}_{x,y}(\kappa_2 s, t, u) = u[s/y] \tag{35}$$

DEFINITION 5.3. Term assignment for $\sqcup, \sqcap$ ([Troelstra 92]).

$$\frac{\Gamma \Rightarrow t : A}{\Gamma \Rightarrow \kappa_1 t : A \sqcup B}\ \sqcup R \qquad\qquad \frac{\Gamma \Rightarrow u : B}{\Gamma \Rightarrow \kappa_2 u : A \sqcup B}\ \sqcup R$$

$$\frac{\Gamma[x : A] \Rightarrow t : C \quad \Gamma[y : B] \Rightarrow u : C}{\Gamma[z : A \sqcup B] \Rightarrow \mathrm{E}^{\sqcup}_{x,y}(z,t,u) : C}\ \sqcup L$$

$$\frac{\Gamma[x : A] \Rightarrow t : C}{\Gamma[z : A \sqcap B] \Rightarrow t[(z)_0/x] : C}\ \sqcap L \qquad \frac{\Gamma[y : B] \Rightarrow t : C}{\Gamma[z : A \sqcap B] \Rightarrow t[(z)_1/y] : C}\ \sqcap L$$

$$\frac{\Gamma \Rightarrow t : A \quad \Gamma \Rightarrow u : B}{\Gamma \Rightarrow \langle t, u \rangle : A \sqcap B}\ \sqcap R$$

In [Morrill 94a], the interpretation for the additive connectives is further refined. The connectives are split up in a semantically *active* version $\sqcap, \sqcup$ with the above Curry-Howard interpretation, and a semantically *neutral* form $\sqcap', \sqcup'$ for the case where one and the same semantic recipe has two distinct syntactic type realizations. The range of application of the semantically neutral form $A \sqcap' B, A \sqcup' B$ is then naturally restricted to the case where $t(A) = t(B)$.

LEXICAL GENERALIZATIONS. Extension of the multiplicative vocabulary with the connectives $\sqcap, \sqcup$ provides type-logical facilities for expressing generalizations with respect to lexical type assignments — a categorial application of the lattice operations originally suggested

[4]Notice that the multiplicative product and additive $\sqcap$ are both interpreted in terms of pairing and projection. As pointed out in §3, categorial term assignment does not insist on an *isomorphism*, but on the weaker homomorphism requirement of the compositionality principle.

in [Lambek 61]. Such uses of the $\sqcap, \sqcup$ vocabulary have been explored for grammar development purposes in [Morrill 94a, Kanazawa 92, Hendriks 95], and in the context of a categorial learning theory in [Adriaans 92], where the meet and join connectives are used in formulating clustering operations of the learning algorithm.

On the most general level the conjunctive $\sqcap$ allows one to compress multiple assignment of types $A_1, \ldots, A_n$ to a vocabulary element into a single assignment of the form $A_1 \sqcap \cdots \sqcap A_n$. But in the case of assignments with shared structure, one can push the generalization further than the top-level conjunction. Given the theorems in (36), one can compress the left-hand side to the more economical right-hand side assignment.

$$(A/C) \sqcap (B/C) \longleftrightarrow (A \sqcap B)/C \qquad (C/A) \sqcap (C/B) \longleftrightarrow C/(A \sqcup B) \tag{36}$$

Notice that the generalization here is to be understood as economy of representation: the double arrow makes clear that on the denotational level, the redundant and the economical type formula are equivalent. Examples of this type of lexical generalization are easy to find. A preposition like *at* takes a noun phrase complement and functions in combination with it both as a nominal modifier (*the house at the corner*) and as a prepositional phrase complement (*laugh at Ferdinand*). The type $(pp \sqcap (n\backslash n))/np$ expresses this. A verb like *know* can be typed as $(np\backslash s)/(np \sqcup s)$: it takes either a sentential or a noun phrase complement to form a verb phrase (*know that he left* versus *know the answer*).

COORDINATION OF UNLIKE CATEGORIES. The grammar logic can exploit the properties of the $\sqcap, \sqcup$ operations in combination with the built-in asymmetry of the derivability relation. Cases of coordination of 'unlike' categories illustrate this type of reasoning in terms of subsumption relations between types. For discussion, see [Morrill 90b, Morrill 94a, Hendriks 95, Bayer & Johnson 95]. Consider the following examples from [Sag e.a. 85]. In (37a) we find coordination of an adjectival complement (*stupid*) with a noun phrase (*a liar*). The (37b) example shows that coordination of unlike categories does not come 'for free': the subcategorizational requirements of, in this case, the copula *be* license a form of coordination which, without this licensing factor, is ungrammatical.

$$\begin{array}{ll} a. & \text{Pat is either stupid or a liar.} \\ b. & \text{*The longwinded and a bully man was my brother.} \end{array} \tag{37}$$

Suppose we assign *be* the type $(np\backslash s)/((n/n) \sqcup np)$. Now the coordination type schema $(X\backslash X)/X$ can be instantiated with $X = (n/n) \sqcup np$, and one can conjoin *stupid and a liar* as like categories. The lexical meaning recipe for the copula can be given in terms of a case construction keying the 'predicational' and the 'identificational' readings to the first and the second injections, respectively, as shown in [Morrill 94a]. An attempt to derive the ungrammatical (37b) by means of a similar disjunctive instantiation of the coordination type schema leads to failure, as shown in (38). Given an instantiation $X = (n/n) \sqcup np$, one can conjoin *longwinded and a bully*, but the disjunctive type cannot be successfully used in the wider context of the noun phrase *the longwinded and a bully man.*

$$\dfrac{\dfrac{}{np/n, n/n, n \Rightarrow np}\,\text{OK} \qquad \dfrac{}{np/n, np, n \Rightarrow np}\,\text{FAIL}}{np/n, (n/n) \sqcup np, n \Rightarrow np}\,\sqcup L \tag{38}$$

FEATURAL DECOMPOSITION. Another suggested application of the Boolean type constructors relates to feature decomposition of categories, cf. [Kanazawa 92, Bayer & Johnson 95].

Suppose we want to distinguish plural and singular nominals. Introduction of new *atomic* types n_{sing} and n_{plur} misses the point that these types share the nominal information, while differing in their number properties. With the additive connectives, we could form the types $n \sqcap sing$ versus $n \sqcap plur$, where $sing$ and $plur$ now are taken to be 'feature' types. The determiner 'a', which requires a singular noun, can be typed $(np \sqcap sing)/(n \sqcap sing)$. The determiner 'the', which combines with either number, is typed simply as np/n.

The additive approach towards featural decomposition has a number of problematic aspects arising from the fact that 'syntactic' categories (like np) and featural information (like $sing$) are treated on a par. Because of the semantic associativity/commutativity of the additives, there is no guarantee that feature types will stay associated with the syntactic categories they are supposed to decorate. Consider an expression like 'her' which is both an accusative pronoun, say $np \sqcap acc \sqcap 3fem$, and a possesive pronoun, $(np/n) \sqcap 3fem$. Putting these two types together with '$\sqcap$', however, we cannot prevent the acc specification to reassociate with np/n. In the sections that follow, we will discuss some proposals that offer a more attractive perspective on the integration of feature information within the type-logical framework.

RECOGNIZING CAPACITY. The effects of additive enrichments of the type language on recognizing capacity have been studied by [Kanazawa 92], see Chapter Twelve for discussion. Unconstrained addition of $\sqcap$ to $\mathbf{L}$ leads beyond context-free recognizing power. But one can constrain the interaction between the multiplicative and the additive vocabulary in such a way that the recognizing power is unaffected. One option is to impose a restriction on the distribution of the additive connectives in lexical assignments to the effect that only the rule of *use* for $\sqcap$ and the rule of *proof* for $\sqcup$ come into play ([Kanazawa 92]). A second option is to restrict the interaction between multiplicatives and additives to the level of atomic types, so that the $\sqcap, \sqcup$ operators never have scope over the multiplicatives ([Dörre & Manandhar 95]). For the grammar writer, such restrictions are not unreasonable.

BOOLEAN RESOURCE MANAGEMENT. As noted at the beginning of this section, the Gentzen rules of Def 5.2 are only approximations of the Boolean intersection/union interpretation for $\sqcap, \sqcup$: the resource sensitive management of the Linear Logic database does not support full Boolean inference. For the distributivity law $A \sqcap (B \sqcup C) \longleftrightarrow (A \sqcap B) \sqcup (A \sqcap C)$, the $\leftarrow$ direction is derivable, but the $\rightarrow$ direction is not, while it is valid for the intersection/union interpretation.

One can resolve this problem in two ways: one can keep the proof theory of Def 5.2 as it is and refine the model theory, so that one indeed obtains completeness; or one can adjust the Gentzen proof rules, so that they are appropriate for the Boolean interpretation of $\sqcap, \sqcup$. For the first alternative, appropriate semantic structures are semilattice ordered groupoids (or 'resource algebras' cf. [Došen 89, Wansing 92b]): structures $\langle W, \cdot, \cap, 1 \rangle$, where W is thought of as a set of information pieces, and where one has a notion of 'information growth' $x \leq y$, defined as $x = x \cap y$. The informational interpretation is quite natural, but it is not fully clear how one could reconcile this with our understanding of $\bullet$ as structural composition.

The second option is to maintain the Boolean interpretation, and make appropriate adjustments to the proof theory. Within the multimodal setting of §4.1, this option can be implemented without 'side-effects' on the interpretation of the multiplicative vocabulary, as has been shown in [Restall 94]. One simply introduces a mode $(\cdot, \cdot)^{\mathbf{2}}$ with Boolean resource management for the family of logical connectives $\sqcap, \sqcup$. The logical rules for $\sqcup$ can be taken

over from Def 5.2 and the logical rules for $\sqcap$ are replaced by the ones given in Def 5.4. These rules are exactly the same as the multiplicative $\bullet$ rules except for the fact that they interpret $\sqcap$ in terms of the Boolean mode $(\cdot,\cdot)^{\mathbf{2}}$. The connectives $\sqcap, \sqcup$ now derive their proper inferential capacities from the structural rule package that characterizes the $(\cdot,\cdot)^{\mathbf{2}}$ mode. Apart from Associativity and Commutativity, this rule package contains mode-restricted Contraction and Weakening.

DEFINITION 5.4. Boolean resource management ([Restall 94]). Logical rules for $\sqcap$. Structural rules for the $(\cdot,\cdot)^{\mathbf{2}}$ mode.

$$\frac{\Delta_1 \Rightarrow A \quad \Delta_2 \Rightarrow B}{(\Delta_1,\Delta_2)^{\mathbf{2}} \Rightarrow A \sqcap B}\sqcap R' \qquad \frac{\Gamma[(A,B)^{\mathbf{2}}] \Rightarrow C}{\Gamma[A \sqcap B] \Rightarrow C}\sqcap L'$$

$$\frac{\Gamma[(\Delta,\Delta)^{\mathbf{2}}] \Rightarrow A}{\Gamma[\Delta] \Rightarrow A}[\mathrm{C}] \qquad \frac{\Gamma[\Delta_1] \Rightarrow A}{\Gamma[(\Delta_1,\Delta_2)^{\mathbf{2}}] \Rightarrow A}[\mathrm{W}]$$

$$\frac{\Gamma[(\Delta_2,\Delta_1)^{\mathbf{2}}] \Rightarrow A}{\Gamma[(\Delta_1,\Delta_2)^{\mathbf{2}}] \Rightarrow A}[\mathrm{P}] \qquad \frac{\Gamma[(\Delta_1,(\Delta_2,\Delta_3)^{\mathbf{2}})^{\mathbf{2}}] \Rightarrow A}{\Gamma[((\Delta_1,\Delta_2)^{\mathbf{2}},\Delta_3)^{\mathbf{2}}] \Rightarrow A}[\mathrm{A}]$$

We leave it as an exercise to show that the earlier $[\sqcap R]$ and $[\sqcap L]$ rules are derivable from $[\sqcap R']$ and $[\sqcap L']$, given the structural options of the $(\cdot,\cdot)^{\mathbf{2}}$ mode. In is shown in [Restall 93] that the Cut rule is admissible in the multimodal system resulting from the combination of the above rules for $\sqcap, \sqcup$ with the standard Lambek-style multiplicatives, and that the system is decidable — the search space can be kept finite, in spite of the presence of the mode-restricted Contraction rule.

EXAMPLE 5.5. Deriving distributivity with Boolean resource management: the direction underivable under the additive regime.

$$\dfrac{\dfrac{\dfrac{\dfrac{A \Rightarrow A \quad B \Rightarrow B}{(A,B)^{\mathbf{2}} \Rightarrow A \sqcap B}\sqcap R'}{(A,B)^{\mathbf{2}} \Rightarrow (A \sqcap B) \sqcup (A \sqcap C)}\sqcup R \qquad \dfrac{\dfrac{A \Rightarrow A \quad C \Rightarrow C}{(A,C)^{\mathbf{2}} \Rightarrow A \sqcap C}\sqcap R'}{(A,C)^{\mathbf{2}} \Rightarrow (A \sqcap B) \sqcup (A \sqcap C)}\sqcup R}{(A,(B \sqcup C))^{\mathbf{2}} \Rightarrow (A \sqcap B) \sqcup (A \sqcap C)}\sqcup L}{A \sqcap (B \sqcup C) \Rightarrow (A \sqcap B) \sqcup (A \sqcap C)}\sqcap L'$$

Unary operators: additive interpretation

In §4.2 we developed a *multiplicative* theory of the unary operators $\Diamond, \Box^{\downarrow}$, presenting them as 'truncated' forms of $\bullet$ and a residual implication. In a language with $\sqcap$, one can develop an alternative *additive* account of the unary control operators. We present the proposals of [Venema 93] for the introduction of a unary operator ∇, decomposable as $A \sqcap Q$, where Q is a type constant picking out a subset of the interpretation domain W — a subset of elements which count as 'special' in a sense indicated by the constant Q. The ∇ operator resolves model-theoretic problems for the 'subalgebra' interpretation of the categorial modalities, which we discuss below.

MODALITIES: SUBALGEBRA INTERPRETATION. The frame semantics for $\Diamond, \Box^{\downarrow}$ interprets the unary vocabulary in terms of a binary accessibility relation that models the relevant notion of multiplicative linguistic composition. The original Edinburgh interpretation for the

$S4$-type $\Box$ operators discussed in §4.2 was presented in terms of groupoid, algebraic models for the multiplicative vocabulary. The semantics for the $\Box$ operator, attributed to Guy Barry in [Morrill 94a], is obtained by distinguishing within the default structural algebra $\langle W, \cdot \rangle$ a *subalgebra* $\langle W_\Box, \cdot \rangle$. The valuation is extended to $\Box$ formulae in the obvious way: $v(\Box A) = v(A) \cap W_\Box$. In the case of the *domain* modalities of [Hepple 90], the elements of $W_\Box$ do not have special resource management properties: one interprets $\Box A$ as the intersection of $v(A)$ with an arbitrary subalgebra. In the case of *structural* modalities for controlled resource management, the groupoid operation '$\cdot$' is governed by an appropriate set of structural equations explicitly restricted to elements of the subalgebra $W_\Box$. See [Morrill 94a] for discussion. For example, in the case of an operator $\Box_p$ for controlled permutation, the relevant equation would be: $\forall x \in W, \forall x' \in W_{\Box_p},\ x \cdot x' = x' \cdot x$.

Unfortunately, the $S4$ proof theory for $\Box$ is incomplete with respect to the intended subalgebra interpretation, for reasons discussed in [Versmissen 93]. The origin of the problem is the condition on the rule of proof $[\Box R]$ which requires that *all* antecedent assumptions be modally decorated. For the subalgebra interpretation of the modalities, this condition is too strong: it rules out sequents that are generally valid. Compare the following sequents.

$$\begin{array}{lll} a. & \mathbf{L}{+}\Box \vdash & \Box B/A, A \Rightarrow C/(A\backslash((\Box B/A)\backslash C)) \\ b. & \mathbf{L}{+}\Box \not\vdash & \Box B/A, A \Rightarrow \Box(C/(A\backslash((\Box B/A)\backslash C))) \end{array} \tag{39}$$

While (39a) is derivable, as the reader can check, (39b) is not, but it *is* generally valid. Pick arbitrary x, y with $x \in v(\Box B/A)$ and $y \in v(A)$. Derivability of (a) and soundness gives (†) $xy \in v(C/(A\backslash((\Box B/A)\backslash C)))$. But we know from the definition of v for the connective '/' that (‡) $xy \in v(\Box B) = v(B) \cap W_\Box \subseteq W_\Box$. The combination of (†,‡) tells us that $xy \in v(C/(A\backslash((\Box B/A)\backslash C))) \cap W_\Box$. But by the definition of v for $\Box$ this means that $xy \in v(\Box(C/(A\backslash((\Box B/A)\backslash C))))$. It is clear why $[\Box R]$ causes the problem. If one could start the derivation of (39b) with $[\Box R]$, one obtains the derivable (39a) as the premise. But the condition on $[\Box R]$ prevents this: the A assumption is not modal.

ADDITIVE INTERPRETATION OF THE STRUCTURAL OPERATORS. The logic for the unary additive operator ∇ consists of three parts. First, there is the usual set of logical rules for use and proof of formulas ∇A. Secondly, the $[Q\circ]$ rule expresses the fact that the subset W_Q of the domain of interpretation is multiplicatively closed, i.e. that we deal with a subalgebra $\langle W_Q, \cdot \rangle$. (Notice that the rule $[Q\circ]$ is in fact a compiled Cut. But being a Cut on a constant, it does not threaten decidability.) Finally, there is a set of ∇-controlled structural rules. Controlled Permutation is given as an example.

DEFINITION 5.6. The additive structural operator ∇ ([Venema 93]). Logical rules, structural rules ($i \in \{1,2\}$). The $[\nabla \mathrm{L}i]$ and $[\nabla \mathrm{R}]$ rules are derivable if one defines ∇A as $A \sqcap Q$.

$$[\nabla\mathrm{L}1]\frac{\Gamma[A] \Rightarrow B}{\Gamma[\nabla A] \Rightarrow B} \quad \frac{\Gamma[Q] \Rightarrow B}{\Gamma[\nabla A] \Rightarrow B}[\nabla\mathrm{L}2] \quad \frac{\Delta \Rightarrow A \quad \Delta \Rightarrow Q}{\Delta \Rightarrow \nabla A}[\nabla\mathrm{R}]$$

$$\frac{\Delta_1 \Rightarrow Q \quad \Delta_2 \Rightarrow Q \quad \Gamma[Q] \Rightarrow A}{\Gamma[(\Delta_1, \Delta_2)] \Rightarrow A}[Q\circ] \qquad \frac{\Delta_i \Rightarrow Q \quad \Gamma[(\Delta_2, \Delta_1)] \Rightarrow A}{\Gamma[(\Delta_1, \Delta_2)] \Rightarrow A}[Q\mathrm{P}]$$

On the basis of the fixed core of logical rules [∇ L], [∇ R], one can design variants by modifying the other rules. The rule $[Q\circ]$ is appropriate in situations where putting together resources which are 'special' in the 'Q' sense produces a result which is also special. One

can easily think of applications where putting together Q resources does not produce a new Q resource. (Using the Q constants for prosodic sorting, for example, the composition of two prosodic words might yield a phrase.) A second source of variation is the ∇ controlled structural rules. As presented above, the rules require *one* subterm of a structurally affected configuration to have the Q property — the subterm which then qualifies as the licensing factor for structural relaxation. Alternatively, one can formulate a version of the structural rules where relaxation requires *all* the immediate subterms of the target configuration to derive Q. See [Versmissen 96] for discussion.

CUT ELIMINATION, COMPLETENESS, ADDITIVE EMBEDDINGS. Cut Elimination, and completeness of the ∇, Q extended language with respect to the subalgebra interpretation in semilattice ordered groupoids is proved in [Venema 93]. Venema also establishes general embedding results showing that the ∇ operator and the associated type constant Q allow one to fully recover the expressivity of a logic $\mathcal{L}$ with a more relaxed resource management regime from within a system $\mathcal{L}'$ with a more stringent form of structure sensitivity. The additive style of structural relaxation can be compared with the 'licensing' direction of the multiplicative embeddings discussed in §4.2. The existence of these alternative approaches suggests a more general question for future research: How does one draw the line between 'multiplicative' and 'additive' aspects of the grammatical composition relationΓ

5.2 Dependent types, type schemata

We move now to a first-order polymorphic type language where 'predicational' basic types take the place of propositional atoms, and to second-order polymorphic type schemata.

5.2.1 First-order polymorphism

The additive view on 'featurally' complex category formulas considers the basic types as unanalysed propositional atoms. But featural information can also be incorporated in the type system by moving from a propositional basis to a first-order predicational system. This view on categorial fine-structure is characteristic of the 'unification-based' versions of categorial grammar, which will be discussed more fully in §6, the section on hybrid architectures. In the type-logical setting proper, predicational types are discussed in [Morrill 94a]. See also [Carpenter 92b], where the basic clausal structure of English is presented in terms of first-order basic type formulae.

PREDICATIONAL BASIC TYPES. Instead of the propositional basic types (such as np for noun phrases) we now take the atomic types to be predicates, and we introduce terms built up from feature constants and feature variables (and maybe feature function symbols) to fill the argument positions of the predicational types. For example: a noun phrase of feminine gender could be typed as $np(fem)$; the type $np(3(fem))$ could serve for feminine noun phrases of the third person, etc. The Gentzen rules for dependent types are given in Def 5.7. In symmetry with the treatment of $\sqcap, \sqcup$, [Morrill 94a] presents semantically inactive variants $\forall', \exists'$ in addition to the semantically active ones below.

DEFINITION 5.7. Dependent function types ($\forall$), dependent sum types ($\exists$) ([Morrill 94a]).

$$\frac{\Gamma[x : A[t/v]] \Rightarrow u : C}{\Gamma[y : \forall vA] \Rightarrow u[y(t)/x] : C}\ \forall L \qquad \frac{\Gamma \Rightarrow t : A}{\Gamma \Rightarrow \lambda v.t : \forall vA}\ \forall R\ (v \text{ not free in } \Gamma)$$

$$\frac{\Gamma \Rightarrow u : A[t/v]}{\Gamma \Rightarrow \langle t, u\rangle : \exists v A}\ \exists R \qquad \frac{\Gamma[x : A] \Rightarrow u : B}{\Gamma[z : \exists v A] \Rightarrow u[\pi_2 z/x] : B}\ \exists L\ (v \text{ not free in } \Gamma,\ B)$$

ILLUSTRATION: UNIFICATION/GENERALISATION THROUGH QUANTIFICATION. To express generalizations over feature information, we can now use the feature variables, and explicitly quantify over these. The definite determiner 'the', for example, could be typed as $\forall N.np(N)/n(N)$, where N is a variable over the number features. The proof rules for the universal quantifier then guarantee combination with singular as well as plural nominal arguments. In [Morrill 94a] it is shown how unification, underspecification and re-entrancy can be expressed type-logically through explicit universal quantification.

But the type-logical reanalysis is not just expressively equivalent to the conventional unification alternative. In the type-logical setting, the essential duality between universal and existential quantification suggests the possibility of assigning types with *existentially* bound feature variables. A problematic empirical area where existential binding seems to be required is the coordination of expressions with clashing feature specifications, e.g. 'boys and girls' where an $np(masc)$ and an $np(fem)$ have to be combined. Unification fails, but under the existential reading $\exists G.np(G)$ the conjuncts can be assigned a common type. What this example suggests is that explicit quantificational type assignments make it possible to trigger both unification and *generalization* as components of the process of linguistic composition.

5.2.2 Second-order polymorphic systems

In the *second order* polymorphic systems, the type language is extended with quantification over *type* variables. Def 5.8 gives the sequent rules for **L**2, the second order polymorphic extension of **L**. For term assignment, one can draw on the constructs of the second order polymorphic λ calculus ([Girard e.a. 89]).

DEFINITION 5.8. Second order quantification in **L**2 ([Emms 93a]). The side condition on $(\forall R)$ and $(\exists L)$ prohibits Y to be free in the conclusion, X in $QY.A$ (where Q is $\forall$ or $\exists$). Under these conditions $QX.A[X/Y]$ can count as an alphabetic variant of $QY.A$.

$$\frac{\Gamma, A[B/X], \Delta \Rightarrow C}{\Gamma, \forall X.A, \Delta \Rightarrow C}\ \forall L \qquad \frac{\Gamma \Rightarrow A}{\Gamma \Rightarrow \forall X.A[X/Y]}\ \forall R(\star)$$

$$\frac{\Gamma \Rightarrow A[B/X]}{\Gamma \Rightarrow \exists X.A}\ \exists R \qquad \frac{\Gamma, A, \Delta \Rightarrow C}{\Gamma, \exists X.A[X/Y], \Delta \Rightarrow B}\ \exists L(\star)$$

CHAMELEON WORDS. Linguistic motivation for second order polymorphism comes from expressions with a chameleon-like behaviour. Coordinating particles *and, or* are cited by Lambek [Lambek 58] as prime examples. Resorting to multiple type assignment for the various manifestations of coordination would clearly miss a generalization. Rather, one would like to assign these particles a type schema $\forall X.(X\backslash X)/X$ generalizing over an infinite family of types. The type schema can then adapt to the context it finds itself in. Besides coordination particles, one can think here of negation $\forall X.X/X$, generalized quantifiers $\forall X.X/(np\backslash X)$, $\forall X.(X/np)\backslash X$, relative pronouns $\forall X.((n\backslash n)/(X\backslash s))/(X/np)$. Polymorphic accounts of these phenomena, and others, have been developed by [Emms 93b], in the context of **L**2. An illustration of the polymorphic account of relative clause formation is given

in Ex 5.9 (where r abbreviates $n\backslash n$). Notice that the reasoning about the type variables exploits the associativity of the underlying base logic: the relative clause body ('s missing an np') is broken in two pieces at the gap site — the part up to the gap is analysed as X/np, the part after the gap site as $X\backslash s$.

EXAMPLE 5.9. Relativization with type assignment $\forall X.(r/(X\backslash s))/(X/np)$ for the relative pronoun ([Emms 93a]). Cf. '(the mathematician) whom Ferdinand considered intelligent'.[5]

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\&c}{np,((np\backslash s)/ap)/np,np,ap\Rightarrow s}}{np,((np\backslash s)/ap)/np,np\Rightarrow s/ap}\,/R}{np,((np\backslash s)/ap)/np\Rightarrow (s/ap)/np}\,/R \quad \dfrac{\dfrac{\dfrac{\&c}{s/ap,ap\Rightarrow s}}{ap\Rightarrow (s/ap)\backslash s}\,\backslash R \qquad r\Rightarrow r}{r/((s/ap)\backslash s),ap\Rightarrow r}\,/L}{(r/((s/ap)\backslash s))/((s/ap)/np),np,((np\backslash s)/ap)/np,ap\Rightarrow r}\,/L}{\forall X.(r/(X\backslash s))/(X/np),np,((np\backslash s)/ap)/np,ap\Rightarrow r}\,\forall L$$

DERIVATIONAL VERSUS VARIABLE POLYMORPHISM. The way we have presented things above, the grammar writer will have recourse to polymorphic type assignment in cases where the built-in notion of type *derivability* of the system one is working with fails to capture a relevant generalization. There is a trade-off here between the degree of sophistication of the multiplicative vocabulary and the work-load one puts on polymorphic type schemata — between derivational and 'variable' polymorphism, to use the terminology of [van Benthem 88a]. Our discussion in the previous sections provides illustrations of the tension in question. In the case of coordination, deriving verb phrase ($np\backslash s$) from sentential s coordination is beyond the reach of a Contraction-free grammar logic: generalized coordination then would qualify as genuinely polymorphic. But consider the case of generalized quantifier expressions. One can adopt a conservative attitude with respect to the multiplicative vocabulary and restrict this to the implicative operators of **L**. The inferential capacity of this system, as we have seen, is too limited to derive the appropriate range of scopal possibilities. In a polymorphic extension **L**2 with double assignment $\forall X.X/(np\backslash X)$, $\forall X.(X/np)\backslash X$ to the generalized quantifiers one can overcome these limitations, as shown in [Emms 93b]. Alternatively, one can move to a more articulate multiplicative language, and introduce a binding operator $q(A,B,C)$ (or the deconstruction of (22) in terms of wrapping). With this more refined vocabulary, the instantiations of the polymorphic type schemata for the generalized quantifier expressions are reduced to derivable type transitions.

The proper division of labour between multiplicative and polymorphic enhancements of the inferential capacity of the grammar logic cannot be decided *a priori* on grounds of semantic or structural expressivity. In fact, Def 5.10 offers a systematic choice between the multiplicative and polymorphic strategies in terms of polymorphic *simulations* of multiplicative extensions of **L**. ($A\uparrow B$ and $A\downarrow B$ are the extraction and infixation connectives of [Moortgat 88], which, in the terminology of §4.2, can themselves be synthesized as $A/\Diamond\Box^{\downarrow}B$ and $\Diamond\Box^{\downarrow}(A/B)$, respectively, for a controlled Permutation $\Diamond$.)

DEFINITION 5.10. Polymorphic simulation of multiplicative extensions of **L** ([Emms 94b]).

p'	$=$	p		$(A\uparrow B)'$	$=$	$\exists X.((X/B')\bullet(X\backslash A'))$	(extraction)
$(A/B)'$	$=$	A'/B'		$(A\downarrow B)'$	$=$	$\forall X.((X/A')/X\backslash B'))$	(infixation)
$(B\backslash A)'$	$=$	$B'\backslash A'$		$q(A,B,C)'$	$=$	$\forall X.(((X/A')\backslash C')/(X\backslash B'))$	('in situ' binding)

[5] In order to cover cases of *peripheral* extraction with this relative pronoun type assignment, one has to lift the **L** prohibition of empty antecedents.

FORMAL PROPERTIES. General logical properties of **L**2 have been studied in [Emms & Leiss 93, Emms 95, Emms 94c]. On the model-theoretic level, [Emms 94c] investigates under what conditions one can obtain completeness results for **L**2 with respect to the ternary relational models, and their specialisations, residuated semigroups and free semigroups (string models). Proof-theoretically, [Emms & Leiss 93] show that the polymorphic system, both in its unrestricted form and with a restriction to outermost-only quantification, enjoys Cut Elimination. To establish the result for unrestricted second order quantifiers is more difficult, since the original Cut Elimination strategy with its induction on the complexity of the Cut formula cannot be used: the quantifier inferences may increase complexity. Still, the *resource sensitivity* of the categorial logic makes it possible to proceed by induction on the proof size.

In the case of the generalized Lambek calculi we have studied in §4.1, Cut Elimination went hand in hand with decidability. For **L**2 this is not the case. Undecidability of **L**2 has been established in [Emms 95] via a generalization of a recent result by [Lincoln e.a. 95]. These authors show that second order intuitionistic propositional logic ($LJ2$) — a system which is known to be undecidable — can be embedded into the multiplicative fragment of second order intuitionistic Linear Logic ($IMLL2$), i.e. **LP**2. The key idea of the embedding is to reintroduce the structural rules of Contraction and Weakening that differentiate between $LJ2$ and $IMLL2$ in the shape of second order $IMLL2$ *formulae* $\forall X.X \multimap (X \otimes X)$ and $\forall X.X \multimap I$. Emms extends this strategy to the structural rule that differentiates between $IMLL2$ and **L** — the rule of Permutation, which is reintroduced via the second order formula $\forall X \forall Y.((X \bullet Y)/X)/Y$.

The undecidability result suggests a general question in relation to the embeddings in §4.2.3: could one design a uniform embedding strategy based on second-order encoding of structural postulates such that the undecidability of $IMLL2$ would carry over to the polymorphic systems of the full sublinear landscape? The question remains open for the time being.

PARSING, RECOGNIZING CAPACITY. The undecidability result for polymorphic **L** provides additional motivation for the search for restricted forms of categorial polymorphism with more pleasant computational properties. For a discussion of the options, see [Barendregt 92]. Emms [Emms 93a] has explored this issue in a discussion of parsing with **L**2. It turns out that for the actual linguistic uses of type schemata mentioned above, one can do with a very mild form of polymorphism where the universal quantifier is restricted to outermost position. Such a restricted version is closely related to the extension of Lambek Calculus with a Substitution Rule, discussed in [van Benthem 88a], and used in [Moortgat 88] in a resolution-based treatment of generalized coordination. Emms [Emms 93a] provides a terminating parsing algorithm for restricted **L**2. With respect to recognizing power, [Emms 94a] shows that extraction covering extension of **L**, such as obtained via the $\uparrow$ operator of Def 5.10 or its **L**2 translation, have recognizing capacity beyond context-free.

6 Hybrid architectures

6.1 Combining type logic and feature logic

The additive type constructors and first or second order quantifiers of §5 offer a purely type-logical perspective on a number of issues that have played a central role in the development of unification formalisms. In this section, we consider the alternative approach of *hybrid*

grammar architectures: frameworks based on an effective mixture of the categorial and unification formalisms. In the development of these hybrid frameworks, one can distinguish two phases.

'First generation' systems: Categorial Unification Grammar (CUG, [Uszkoreit 86, Bouma 88a]) and Unification Categorial Grammar (UCG, [Calder e.a. 88]). These systems were introduced a decade ago, during the initial wave of enthusiasm for unification-based grammar architectures. From a taxonomic perspective, CUG and UCG are unification grammars with a highly restricted categorial rule base (typically Functional Application, plus maybe some other rule schemata), and a categorially inspired make-up of the basic declarative unit, the 'sign'.

'Second generation' systems. A type-logical feature which the first generation systems lack is the ability for fully general hypothetical reasoning. Within the context of Gabbay's program for 'combining logics' [Gabbay 94], a number of proposals have recently been put forward for a genuine integration of type logic and feature logic. These second generation systems are designed so as to fully preserve the inferential capacities of the constituting logics. They offer a grammar logic with a categorial treatment of grammatical composition and its resource management, combined with a typed constraint-based account of internal categorial fine-structure. Exponents of this type of integration are [Dörre & Manandhar 95] and [Dörre e.a. 94].

First generation systems

The starting point for the CUG/UCG hybrids is an encoding of categorial type formulae in the attribute-value language of unification grammar. Such encodings can be implemented in a variety of ways. Ignoring matters of economy, one could use (following [Bouma 88b]) a feature CAT with atomic values for basic categories, or attribute-value specifications with attributes VAL (value), DIR (directionality), ARG (argument), in the case of compound (implicational) categories. In (40a) one finds the encoding of a transitive verb type $(np\backslash s)/np$, displayed as an attribute-value matrix. The category specification can be refined with phonological, semantic, or morphophonological information, by adding specifications for features PHON, SEM, AGR in addition to CAT, as in the (40b) specification for a third person singular noun phrase 'Ferdinand'.

$$a.\ \begin{bmatrix} \text{CAT} : \begin{bmatrix} \text{VAL} : \begin{bmatrix} \text{CAT} : \begin{bmatrix} \text{VAL} : \begin{bmatrix} \text{CAT} : s \end{bmatrix} \\ \text{DIR} : \textit{left} \\ \text{ARG} : \begin{bmatrix} \text{CAT} : np \end{bmatrix} \end{bmatrix} \end{bmatrix} \\ \text{DIR} : \textit{right} \\ \text{ARG} : \begin{bmatrix} \text{CAT} : np \end{bmatrix} \end{bmatrix} \end{bmatrix} \qquad b.\ \begin{bmatrix} \text{CAT} : np \\ \text{PHON} : \textit{'Ferdinand'} \\ \text{SEM} : f \\ \text{AGR} : \begin{bmatrix} \text{NUM} : \textit{sing} \\ \text{PER} : 3 \end{bmatrix} \end{bmatrix} \tag{40}$$

Turning to the rule schemata of these systems, the basic law of categorial combination, Functional Application, is combined with the basic principle for fusing conglomerates of featural information: unification. In the Application reduction $A/B \bullet B' \rightarrow A'$, the types B, B', now considered as feature structures, are required to be unifiable rather than identical. Because there may be structure sharing between the A and B components of the functor A/B, the result of the combination A' may be affected by the unification of B and B' — a

feature which is fully exploited to build up complex semantic or phonological representations via unification.

The unification-based extensions of categorial grammar, as appears from the above, rely on feature logic in two distinct areas: one concerns the internal informational make-up of signs, the other relates to their external combinatory potential. We discuss these two aspects in turn.

CATEGORIAL FINE-STRUCTURE AND LEXICAL ORGANISATION. First and foremost, the language of feature logic can be used as a tool to obtain a decomposition of the crude 'atomic' category symbols of standard categorial grammar. Such decomposition allows for the expression of fine-grained distinctions in individual lexical entries, and of categorial generalizations relating classes of entries. We have seen that grammatical macro structure, in the categorial frameworks, is fully projected from the lexicon. With such a central role assigned to the lexicon, it is of crucial importance for realistic grammar development that the huge amounts of lexical information are *organized* in an efficient way. Within unification-based formalisms it has been customary to think of the lexicon as being structured using *templates* (or *macros*), cf. [Shieber 86]. A template contains the information relevant for a class of lexical entries. Instead of fully specifying the information associated with a lexical entry, one assumes that the entry can 'inherit' from one or more templates (where inheritance amounts to (non-monotonic) unification). Thus, large amounts of information may be represented succinctly and a wide range of linguistic generalizations about the lexicon can be captured. Such aspects of lexical organization have been studied in depth in [Bouma 88a].

A second aspect of lexical organization concerns the *lexical rule* component. Lexical rules can capture relations between type assignments that are not covered by derivational laws. There is a question of balance here between the inferential capacity of the derivational system and the work load put on the lexical rule component. The inferential power of the derivational engine in the case of the unification-based categorial systems is limited, as we have seen. Consequently, lexical rules play an important role in these systems (as in *HPSG*, [Pollard & Sag 94]). In [Bouma & van Noord 94], lexical polymorphism is implemented via recursive constraints. Techniques of delayed evaluation are employed to reason efficiently with such constraints. For a general assessment of the computational complexity of lexical rules in categorial or unification-based frameworks, the reader can turn to [Carpenter 91], who shows that the type of operations on subcategorization information actually used lead to Turing complexity. This makes the question of the division of labour between derivational polymorphism and lexical rules pregnant again, and shows the need for a restricted *theory* of lexical rules.

UNIFICATION-BASED GRAMMATICAL COMPOSITION. The combination of categorial reduction (Application) with unification makes it possible to rule out illformed expressions via unification clashes. A verb form like *runs*, for example, might be assigned a feature specification $\langle num \rangle sing \wedge \langle person \rangle 3rd$ for its subject argument. Combination with the above entry for *Ferdinand* succeeds, but combination with a subject *they*, with specification $\langle num \rangle plur \wedge \langle person \rangle 3rd$ for its agreement features will fail because of the incompatibility of the constraints for the $\langle num \rangle$ attribute.

Although this picture of a unification-based refinement of categorial reduction works fine in the case of fully specified featural information, it runs into problems when *underspecification* comes into play. These problems have been highlighted in [Bayer & Johnson 95].

The feature structure encoding of categorial formulae, as given above, is not entirely faithful: it does not take into account the *polarity* of the subformulae. Categorial reasoning is sensitive to these polarity distinctions — the monotonicity inferences of Prop 2.18 (3–5) license weakening of positive subformulae and strengthening of negative ones. Unification, being a symmetric operation, is ill-equiped to capture the polarity asymmetries that turn up in type-logical inference. The nature of the conflict can be illustrated with the case of conjunction of unlike categories, cf. (37). Consider the following (partial) lexical specifications (where COMPLEMENT stands for the path CAT | ARG | CAT in the transitive verb attribute-value matrix given above):

$$\textit{grew}: \left[\begin{array}{l}\text{COMPLEMENT}: \left[\begin{array}{l}\text{NOUN}: + \\ \text{VERB}: +\end{array}\right]\end{array}\right] \quad \textit{became}: \left[\begin{array}{l}\text{COMPLEMENT}: \left[\begin{array}{l}\text{NOUN}: +\end{array}\right]\end{array}\right]$$

$$\textit{wealthy}: \left[\begin{array}{l}\text{VERB}: + \\ \text{NOUN}: +\end{array}\right] \qquad \textit{a Republican}: \left[\begin{array}{l}\text{VERB}: - \\ \text{NOUN}: +\end{array}\right] \tag{41}$$

Assume (with [Sag e.a. 85, Shieber 92]) an account of coordination where feature specification for the conjunction as a whole is the *generalization* of the specifications on the conjuncts, i.e. the most specific category which subsumes each conjunct. We can conjoin *wealthy and a Republican* in [NOUN:+], removing the conflicting constraint on the VERB attribute, and derive the well-formed *Kim became wealthy and a Republican.* But on the same account, *grew and remained* are conjoinable as [COMPLEMENT:[NOUN:+]], so that the ill-formed (42) cannot be blocked. Polymorphism here would demand a *conjunctive* effect in antecedents (ranges of lexical functors) but a *disjunctive* effect in succedents (domains of lexical functors). As we have seen in the discussion of (37), this can be achieved in the type-logical setting by having $\sqcap$ and $\sqcup$ (or explicit quantification $\forall$ and $\exists$) systematically alternating according to polarity of context.

* Kim grew and remained wealthy and a republican (42)

Second generation systems

The problems with the combination of unification and categorial derivability have led to the investigation of more delicate mixtures of type logic and feature logic. We briefly discuss the proposals of [Dörre & Manandhar 95] for 'layering' Lambek type logic over a base logic of feature formulas for the description of categorial fine-structure, and the 'fibered' combination of [Dörre e.a. 94] where feature formulas play the role of global constraints over type-logical derivations.

The layered architecture of [Dörre & Manandhar 95] realizes the interface between type logic and feature logic in terms of a *subtyping* discipline on the atomic types $\mathcal{A}$. In [Lambek 68], such an extension of the original calculus is shown to be decidable, cf. [Buszkowski 88] for discussion. The interpretation mapping respects the subtyping pre-order, i.e. $v(p) \subseteq v(q)$ whenever $p \preceq q$ for $p, q \in \mathcal{A}$. On the proof-theoretic level, one refines the Axiom schema to take into account the specificity ordering of the atomic types: $p \Rightarrow q$ holds, provided $p \preceq q$. The canonical model construction of Def 2.4 readily accommodates the subtype ordering, and provides soundness/completeness for the $\preceq$ extensions of the various calculi in the categorial hierarchy.

From the type-logical perspective, the subtyping relation can be regarded as a 'black box'. One obtains the desired combination of type and feature logic by replacing the atomic formulae p, q of the Lambek component by feature-logic formulae ϕ, ψ, and interpreting the subtype ordering $\preceq$ as entailment checking $\phi \models \psi$. The design of the combination illustrates the philosophy of 'zooming-in, zooming-out' for mixed systems described in [Blackburn & de Rijke 94]: as far as grammatical composition is concerned, Lambek type inference proceeds in the ordinary fashion until one reaches the atoms of the Lambek component. At that point, one 'zooms in' and switches to the feature logic which describes the internal fine-structure of these atoms. The interference of unification with categorial derivability is avoided: at the feature logical level, one performs *subsumption checking* rather than unification. In the Application case, a functor A/B can be freely applied to *subtypes* of B, not to supertypes.

Notice that the layered style of combination is modular in the sense that interaction between the type logic and feature logic components is restricted to the level of type-logical atoms. This has pleasant consequences for the formal power of the combination. It is shown in [Dörre & Manandhar 95] that the layered combination with **L** does not affect the context-free recognizing capacity, in contrast with the 'free' interleaving of multiplicatives and additives studied in [Kanazawa 92].

The pleasant modularity of the layered combination is at the same time its limitation: it does not provide the means of expressing featural structure-sharing (re-entrancy) between subtypes of a categorial formula. As we have seen, such re-entrancies are used for the unification-driven construction of semantic or phonological representations. In [Dörre e.a. 94], one finds a proposal for a complementary combination strategy which allows such structure sharing. In this 'fibered' architecture, one associates each atom of the categorial layer with a variable, and then imposes global feature constraints over these variables.

BACKGROUND, FURTHER READING. Decidability aspects of the unification hybrids are discussed in [van Benthem 91,95]. The computational complexity of unification-based formalisms is studied in [Trautwein 95]. Via simulations of CUG the NP-hard lower bound for that system is transfered to other formalisms in the unification-based family. The system LexGram [König 95] implements the hybridization strategies discussed above. Recent developments within $HPSG$ complement the categorially-inspired theory of subcategorization with simulations of principles such as Geach or Lifting, cf. the treatment of complement inheritance in [Hinrichs & Nakazawa 94], or the incorporation of adjuncts into SUBCAT of [Miller 92].

6.2 Labeled Deductive Systems

The program of Labeled Deductive Systems [Gabbay 94] provides a general framework for the design of hybrid logics, based on the technique of labeling. In labeled deduction, each occurrence of a formula is associated with a label yielding the pair $x : A$ with label x and formula A as the basic declarative unit. Correspondingly, consequence relations hold of structures of labeled formulas, so that in place of $A_1, \ldots, A_n \Rightarrow B$, we have $x_1 : A_1, \ldots, x_n : A_n \Rightarrow y : B$. And accordingly, rules of inference characterizing such consequence relations must specify how actions on formulas and actions on associated labels interact.

A familiar example of labeling is the Curry-Howard morphism, discussed in §3. Gabbay [Gabbay 94] recognized the power and flexibility of the seemingly simple step of labeling and generalized its application across a diverse spectrum of logical problems. From this more

abstract perspective, labels represent an extra 'informational resource' which is present in deduction and can play a variety of roles. One obtains a scala of labeling regimes depending on the degree of autonomy between the formula component and the label component of the basic declarative units. At the conservative end of the spectrum, the label simply keeps a passive record of inferential steps. Curry-Howard labeling is a case in point. In this simple situation, the labels provide no feedback to influence the deductive process. A more complex situation arises in cases in which the operations on labels are only partially defined: under these conditions, the labeling system acts as a *filter* on deduction, since an inference step requiring a label operation that cannot be carried out cannot go through. In more complex cases, there is a genuine two-way communication between labels and formulas. Initial investigations of such possibilities can be found in [Gabbay 94],[Blackburn & de Rijke 94].

Within the categorial context, labeled deduction has appeared in a variety of different research directions. One focus of interest has been the use of labels to codify resource-management in different systems of proof, especially in adaptations of Girard's 'proof-nets' to various forms of categorial inference. We discuss this use in §7. Another focus of work on labeled deduction has been the investigation of how various logical systems are related to particular model structures. Already in 1986, [Buszkowski 86] employed λ-term labels in his study of completeness proofs for various members of the Lambek family of substructural logics. An LDS presentation of **NL** in [Venema 94] forms the basis for completeness proofs for tree-based models. We have seen in §2 that the unlabeled system **NL** is incomplete for this type of semantics: labeling, in this case, makes an irreducible contribution to the deductive strength of the logic. In a similar vein, [Kurtonina 95] shows how the simple and multimodal Lambek systems of §2 and §4.1 can be reconstructed in such a way that they share a common system of type-inference for formulas and distinct systems of resource-management involving only the associated labels, and provides completeness proofs for the appropriate ternary-relation frames. This direction of work illustrates a basic slogan of labeled deduction: 'bringing semantics into syntax'. The labels make it possible to discriminate among cases which the formula-language alone cannot tell apart.

The work just described reflects traditional logical questions of proof- and model-theory. But the study of labeled deduction for type-logical grammatical inference has intrinsic motivation as well: natural language (as stressed in [Oehrle 88]) is characteristically *multi-dimensional*, in a way that pairs phonological, syntactic, and interpretive information. Labeling can be used to explicitly manipulate these grammatical dimensions and their complex interactions. We mention some relevant studies. The system of 'Natural Logic' of [Sanchez 95] presents a combined calculus of categorial deduction and natural language inference: the inferential contribution of subexpressions is accounted for in terms of a labeling system decorating Lambek deductions with monotonicity markings. The work on ellipsis and cross-over phenomena of [Gabbay & Kempson 92] similarly exploits the structure of a labeled natural deduction system in order capture the context-dependency of natural language interpretation. This form of labeled deduction, and the systems proposed in [Oehrle 95] in studies of quantification and binding, essentially interleave phonological, syntactic, semantic and pragmatic information, and assign to the labels an active role in controlling and constraining the course of proof. Finally, the general framework of labeled deduction provides a unifying point of view in which to study different proposals concerning the multidimensional nature of linguistic structure, ranging from the sequential architecture of the Government & Binding school to the family of parallel architectures found in the 'sign'-based system of *HPSG*, the correspondences between f-structure, c-structure, and σ-structure of LFG, and

various members of the family of Tree Adjoining Grammars. These systems can be simulated (at least partially) in the framework of labeled type-logical deduction, in a way that may reveal underlying points of similarity and sharpen understanding of points of essential difference. See [Joshi & Kulick 95] for an exploration of this perspective.

7 Categorial parsing as deduction

In this section we turn to the computational study of categorial type logics, and discuss some aspects of their algorithmic proof theory, under the slogan 'Parsing as Deduction' — a slogan which in the type-logical framework assumes a very literal interpretation. In §7.1 we return to a problem that was already signalled in §3: the many-to-one correspondence between Gentzen proofs and λ-term meaning recipes. We discuss the procedural control strategies that have been proposed to remove this source of 'spurious ambiguity' from Gentzen proof search. In §7.2, we present an attractive alternative for Gentzen proof search, inspired by what Girard [Girard 95a] has called the Natural Deduction for resource logics — the 'proof nets' of Linear Logic. In the spirit of §6.2, we show how labeling can be used to implement an appropriate level of control over the linguistic resources.

7.1 Proof normalization in Gentzen calculus

From the literature on automated deduction, it is well known that Cut-free Gentzen proof search is still suboptimal from the efficiency perspective: there may be different (Cut-free!) derivations leading to one and the same proof term. Restricting ourselves to the implicational fragment, the spurious non-determinism in the search space has two causes ([Wallen 90]): (i) permutability of [L] and [R] inferences, and (ii) permutability of [L] inferences among themselves, i.e. non-determinism in the choice of the active formula in the antecedent. A so-called *goal-directed* (or: uniform) search regime performs the non-branching [R] inferences before the [L] inferences (re (i)), whereas *head-driven* search commits the choice of the antecedent active formula in terms of the goal formula (re (ii)). Such optimized search regimes have been proposed in the context of Linear Logic programming in [Hodas & Miller 94, Andreoli 92]. In the categorial setting, goal-directed head-driven proof search for the implicational fragment **L** was introduced in [König 91] and worked out in [Hepple 90] who provided a proof of the safeness (no proof terms are lost) and non-redundancy (each proof term has a unique derivation) of the method. We present the search regime in the format of [Hendriks 93] with Curry-Howard semantic term labelling.

Definition 7.1. Goal-directed head-driven search for product-free **L** ([Hendriks 93]).

$$[\mathrm{Ax}/\star\mathrm{L}]\frac{}{x : p^\star \Rightarrow x : p} \qquad \frac{\Gamma, u : B^\star, \Gamma' \Rightarrow t : p}{\Gamma, u : B, \Gamma' \Rightarrow t : p^\star}[\star\mathrm{R}]$$

$$[/\mathrm{R}]\frac{\Delta, x : B \Rightarrow t : A^\star}{\Delta \Rightarrow \lambda x.t : A/B^\star} \qquad \frac{\Delta \Rightarrow u : B^\star \quad \Gamma, x : A^\star, \Gamma' \Rightarrow t : C}{\Gamma, s : A/B^\star, \Delta, \Gamma' \Rightarrow t[su/x] : C}[/\mathrm{L}]$$

$$[\backslash\mathrm{R}]\frac{x : B, \Delta \Rightarrow t : A^\star}{\Delta \Rightarrow \lambda x.t : B\backslash A^\star} \qquad \frac{\Delta \Rightarrow u : B^\star \quad \Gamma, x : A^\star, \Gamma' \Rightarrow t : C}{\Gamma, \Delta, s : B\backslash A^\star, \Gamma' \Rightarrow t[su/x] : C}[\backslash\mathrm{L}]$$

The above **L*** calculus eliminates the spurious non-determinism of the original presentation **L** by annotating sequents with a procedural control operator '*'. Goal sequents $\Gamma \Rightarrow t : A$ in **L** are replaced by **L*** goal sequents $\Gamma \Rightarrow t : A^\star$. With respect to the first cause of spurious ambiguity (permutability of [L] and [R] inferences), the control part of the [R] inferences forces one to remove all connectives from the succedent until one reaches an atomic succedent. At that point, the '*' control is transmitted from succedent to antecedent: the [*R] selects an active antecedent formula the head of which ultimately, by force of the control version of the Axiom sequent [*L], will have to match the (now atomic) goal type. The [L] implication inferences initiate a '*' control derivation on the minor premise, and transmit the '*' active declaration from conclusion to major (right) premise. The effect of the flow of control information is to commit the search to the target type selected in the [*R] step. This removes the second source of spurious ambiguity: permutability of [L] inferences.

Prop 7.2 sums up the situation with respect to proofs and readings in **L** and **L***. Syntactically, derivability in **L** and **L*** coincide. Semantically, the set of **L*** proof terms forms a subset of the **L** terms. But, modulo logical equivalence, no readings are lost in moving from **L** to **L***. Moreover, the **L*** system has the desired one-to-one correspondence between readings and proofs.

PROPOSITION 7.2. Proofs and readings ([Hendriks 93]).

1. $\mathbf{L}^\star \vdash \Gamma \Rightarrow A^\star$ iff $\mathbf{L} \vdash \Gamma \Rightarrow A$
2. $\mathbf{L}^\star \vdash \Gamma \Rightarrow t : A^\star$ implies $\mathbf{L} \vdash \Gamma \Rightarrow t : A$
3. $\mathbf{L} \vdash \Gamma \Rightarrow t : A$ implies $\exists t', t' = t$ and $\mathbf{L}^\star \vdash \Gamma \Rightarrow t' : A^\star$
4. if π_1 is an **L*** proof of $\Gamma \Rightarrow t : A$ and π_2 is an **L*** proof of $\Gamma \Rightarrow t' : A$ and $t = t'$, then $\pi_1 = \pi_2$

EXAMPLE 7.3. Without the constraint on uniform head-driven search, there are two **L** sequent derivations for the Composition law, depending on the choice of a/b or b/c as active antecedent type. They produce the same proof term. Of these two, only the first survives in the **L*** regime.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{(c)^\star \Rightarrow c}\star L}{c \Rightarrow (c)^\star}\star R \quad \dfrac{}{(b)^\star \Rightarrow b}\star L}{(b/c)^\star, c \Rightarrow b}/L}{b/c, c \Rightarrow (b)^\star}\star R \qquad \dfrac{}{(a)^\star \Rightarrow a}\star L}{(a/b)^\star, b/c, c \Rightarrow a}/L}{a/b, b/c, c \Rightarrow (a)^\star}\star R}{a/b, b/c \Rightarrow (a/c)^\star}/R$$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{(c)^\star \Rightarrow c}\star L}{c \Rightarrow (c)^\star}\star R \quad \dfrac{\text{FAIL}}{a/b, (b)^\star \Rightarrow a}}{a/b, (b/c)^\star, c \Rightarrow a}/L}{a/b, b/c, c \Rightarrow (a)^\star}\star R}{a/b, b/c \Rightarrow (a/c)^\star}/R$$

UNIFORM PROOF SEARCH: MODAL CONTROL. The control operators $\Diamond, \Box^\downarrow$ make it possible to enforce the König-Hepple-Hendriks uniform head-driven search regime via a modal translation, as shown in [Moortgat 95]. This illustrates a second type of control that can be logically implemented in terms of the unary vocabulary: a procedural/dynamic form of control rather than the structural/static control of §4.2.3. The $\Diamond, \Box^\downarrow$ annotation is a variation on the "lock-and-key" method of [Lincoln e.a. 95]: one forces a particular execution strategy for successful proof search by decorating formulae with the $\Box^\downarrow$ ('lock') and $\Diamond$

('key') control operators. For the selection of the active formula, one uses the distributivity principes $K1, K2$, in combination with the base residuation logic for $\Diamond, \Box^{\downarrow}$. To establish the equivalence with **L*** search, one can use the sugared presentation of **L** where Associativity is compiled away so that *binary* punctuation $(\cdot,\cdot)$ can be omitted (but not the unary $(\cdot)^{\Diamond}$!). This gives the following compiled format for $K1, K2$:

$$\frac{\Gamma, (A)^{\Diamond}, \Gamma' \Rightarrow B}{(\Gamma, A, \Gamma')^{\Diamond} \Rightarrow B}\ K' \tag{43}$$

The mappings (44) $(\cdot)^1, (\cdot)^0 : \mathcal{F}(/,\backslash) \mapsto \mathcal{F}(/,\backslash,\Diamond,\Box^{\downarrow})$, for antecedent and succedent formula occurrences, respectively, are defined as follows.

$$\begin{array}{ll} (p)^1 = p & (p)^0 = \Box^{\downarrow} p \\ (A/B)^1 = (A)^1/(B)^0 & (A/B)^0 = (A)^0/\Box^{\downarrow}(B)^1 \\ (B\backslash A)^1 = (B)^0\backslash(A)^1 & (B\backslash A)^0 = \Box^{\downarrow}(B)^1\backslash(A)^0 \end{array} \tag{44}$$

We have the following proposition ([Moortgat 95]).

$$\mathbf{L}^{\star} \vdash \Gamma \Rightarrow A^{\star} \qquad \text{iff} \qquad \mathbf{L}\Diamond\mathbf{K}' \vdash \Box^{\downarrow}(\Gamma)^1 \Rightarrow (A)^0 \tag{45}$$

Example 7.4. Uniform head-driven search: modal control. We illustrate how a wrong identification of the antecedent head formula leads to failure. Below the modal translation of the crucial upper part of the failing **L*** derivation in Ex 7.3.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{c \Rightarrow c}{(c)^1 \Rightarrow c}\,(\cdot)^1}{(\Box^{\downarrow}(c)^1)^{\Diamond} \Rightarrow c}\,\Box^{\downarrow}L}{\Box^{\downarrow}(c)^1 \Rightarrow \Box^{\downarrow}c}\,\Box^{\downarrow}R}{\Box^{\downarrow}(c)^1 \Rightarrow (c)^0}\,(\cdot)^0 \qquad \dfrac{\text{FAILS}}{\Box^{\downarrow}(a/b)^1, (b)^1 \Rightarrow a}\,\dagger}{\Box^{\downarrow}(a/b)^1, (b)^1/(c)^0, \Box^{\downarrow}(c)^1 \Rightarrow a}\,/L}{\Box^{\downarrow}(a/b)^1, (b/c)^1, \Box^{\downarrow}(c)^1 \Rightarrow a}\,(\cdot)^1}{\Box^{\downarrow}(a/b)^1, (\Box^{\downarrow}(b/c)^1)^{\Diamond}, \Box^{\downarrow}(c)^1 \Rightarrow a}\,\Box^{\downarrow}L}{(\Box^{\downarrow}(a/b)^1, \Box^{\downarrow}(b/c)^1, \Box^{\downarrow}(c)^1)^{\Diamond} \Rightarrow a}\,K'}{\Box^{\downarrow}(a/b)^1, \Box^{\downarrow}(b/c)^1, \Box^{\downarrow}(c)^1 \Rightarrow \Box^{\downarrow}a}\,\Box^{\downarrow}R$$

Consider first the interaction of [/R] rules and selection of the active antecedent type. Antecedent types all have $\Box^{\downarrow}$ as main connective. The $\Box^{\downarrow}$ acts as a *lock*: a $\Box^{\downarrow}A$ formula can only become active when it is *unlocked* by the key $\Diamond$ (or $(\cdot)^{\Diamond}$ in structural terms). The key becomes available only when the head of the goal formula is reached: through residuation, $[\Box^{\downarrow}R]$ transmits $\Diamond$ to the antecedent, where it selects a formula via $[K']$. There is only *one* key $\Diamond$ by residuation on the $\Box^{\downarrow}$ of the goal formula. As soon as it is used to unlock an antecedent formula, that formula has to remain active and connect to the Axiom sequent. In the derivation above, the key to unlock $\Box^{\downarrow}(a/b)^1$ has been spent on the wrong formula. As a result, the implication in $(a/b)^1$ cannot become active.

The normalization strategy for the implicational fragment can be extended to cover the $\bullet$ connective as well, as shown in Andreoli's 'focusing proofs' approach for **LP** (the

multiplicative fragment of Linear Logic). But in the $\bullet$ case, a residue of 'spurious' non-determinism remains. Summarizing the above, we see that the Gentzen format has certain limitations as a vehicle for efficient categorial computation. To overcome these limitations, the above proposals *increase* the Gentzen bookkeeping by adding procedural control features. The alternative is to switch to a data structure for categorial derivations which effectively removes the sources of computational inefficiency. We turn to such an alternative below.

7.2 Proof nets and labeled deduction

In Sections 4.1 and 4.2 we have described the shift from the study of individual categorial systems to the mixed architecture of multimodal type logics. In this section, we consider the mixed architecture from a computational point of view. The central objective in this area is to develop a *general* algorithmic proof theory for the multimodal grammar logics. This is an active area of research (see a.o. [Moortgat 92, Morrill 95c, Hepple 94, Oehrle 95]) with an evolving methodology centering around the combination of the techniques of proof nets and labeled deduction, i.e. resolution theorem proving over labeled literals (rather than Gentzen style proof search over structured formula databases). Here are some desiderata that guide current work. In practice, one finds the expected tension between generality/expressivity and efficiency.

–soundness and completeness of the labeling regime w.r.t. the interpretation;
–expressivity: support for the full language $\Diamond, \Box^{\downarrow}, /, \bullet, \backslash$;
–modular treatment of 'logic' (residuation) and 'structure' (resource management);
–reversibility: neutrality w.r.t. parsing and generation;
–efficient compilation techniques.

In Linear Logic, [Girard 87] has advocated 'proof nets' as the appropriate proof-theoretic framework for resource logics. The proof net approach has been studied in the context of categorial type logics in [Roorda 91]. In this section, we first discuss semantic λ-term labelling for categorial proof nets. On the basis of the semantic labelling one can characterize the appropriate well-formedness conditions for proof nets and the correspondence between nets and **LP** sequent derivations. In order to capture the syntactic fine-structure of systems more discriminating than **LP**, and multimodal architectures with interacting unary and binary multiplicatives, we complement the semantic labeling with *structure* labeling. Both types of labeling 'bring semantics into the syntax', as the slogan has it: for the structure labeling, this is the semantics for the form dimension of the linguistic resources, as discussed in §2.

SEMANTIC LABELING. Building a proof net corresponding to a sequent $\Gamma \Rightarrow A$ is a three stage process. The first stage is deterministic and consists in unfolding the formula decomposition tree for the A_i antecedent terminal formulae and for the goal formula A. The unfolding keeps track of the antecedent/succedent occurrence of subformulae: we distinguish $(\cdot)^1$ (antecedent) from $(\cdot)^0$ (succedent) unfolding, corresponding to the sequent rules of use and proof for the connectives. We call the result of the unfolding a *proof frame*. The second stage, corresponding to the Axiom case in the Gentzen presentation, consists in linking the literals with opposite signature. We call an arbitrary linking connecting the leaves of the proof frame a *proof structure*. Not every proof structure corresponds to a sequent derivation. The final stage is to perform a wellformedness check on the proof structure graph in

order to identify it as a *proof net*, i.e. a structure which effectively corresponds to a sequent derivation.

DEFINITION 7.5. Formula decomposition. Antecedent unfolding $(\cdot)^1$, succedent unfolding $(\cdot)^0$. $\exists$-type ($\Diamond$L,$\bullet$L, /R,\R) and $\forall$-type ($\Box^{\downarrow}$L,/L,\L,$\bullet$R) decomposition.

$$\frac{(A)^1 \quad (B)^0}{(A/B)^1}\,\forall \qquad \frac{(B)^1 \quad (A)^0}{(A/B)^0}\,\exists \qquad \frac{(B)^0 \quad (A)^1}{(B\backslash A)^1}\,\forall \qquad \frac{(A)^0 \quad (B)^1}{(B\backslash A)^0}\,\exists$$

$$\frac{(A)^1 \quad (B)^1}{(A \bullet B)^1}\,\exists \qquad \frac{(B)^0 \quad (A)^0}{(A \bullet B)^0}\,\forall$$

$$\frac{(A)^1}{(\Diamond A)^1}\,\exists \qquad \frac{(A)^0}{(\Diamond A)^0}\,\forall \qquad \frac{(A)^1}{(\Box^{\downarrow} A)^1}\,\forall \qquad \frac{(A)^0}{(\Box^{\downarrow} A)^0}\,\exists$$

For the checking of the well-formedness conditions, there are various alternatives, such as Girard's original 'long trip' condition, or the coloring algorithm of [Roorda 91]. Here we develop a labeling approach, because it is naturally adaptable to the linguistic application of the type logics in parsing and generation. The purpose of the labeling regime in this case is to push all relevant information about the structure of the proof frame up to the atomic leaf literals, so that the wellformedness check can be formulated locally in terms of resolution on the axiom links.

DEFINITION 7.6. Labelled formula decomposition. Positive (antecedent) unfolding $(\cdot)^1$, negative (succedent) unfolding $(\cdot)^0$. We use x, y, z (t, u, v) for object-level variables (terms), M, N for meta-level variables. Newly introduced object-level variables and metavariables in the rules below are chosen fresh.

$$\text{Axiom links} \quad \frac{}{t : (A)^1 \quad M : (A)^0} \qquad \frac{}{M : (A)^0 \quad t : (A)^1} \quad \text{with } M := t$$

$$\frac{t(M) : (A)^1 \quad M : (B)^0}{t : (A/B)^1} \qquad \frac{x : (B)^1 \quad N : (A)^0}{\lambda x.N : (A/B)^0}$$

$$\frac{M : (B)^0 \quad t(M) : (A)^1}{t : (B\backslash A)^1} \qquad \frac{N : (A)^0 \quad x : (B)^1}{\lambda x.N : (B\backslash A)^0}$$

$$\frac{(t)_0 : (A)^1 \quad (t)_1 : (B)^1}{t : (A \bullet B)^1} \qquad \frac{N : (B)^0 \quad M : (A)^0}{\langle M, N \rangle : (A \bullet B)^0}$$

$$\frac{{}^{\cup}t : (A)^1}{t : (\Diamond A)^1} \qquad \frac{M : (A)^0}{{}^{\cap}M : (\Diamond A)^0} \qquad \frac{{}^{\vee}t : (A)^1}{t : (\Box^{\downarrow} A)^1} \qquad \frac{M : (A)^0}{{}^{\wedge}M : (\Box^{\downarrow} A)^0}$$

For the binary vocabulary, the wellformedness conditions of Def 7.7 identify **LP** proof nets among the proof structures. Roorda also shows that one can narrow this class to the **L** proof nets by imposing an extra *planarity* constraint forbidding 'crossing' axiom linkings

(interpreting the formula decomposition steps of Def 7.6 in an order-sensitive way). Evaluating this proposal, [Hendriks 93] observes that don't care non-determinism is removed at the declarative level, *not* at the algorithmic/procedural level: the conditions act as filters, in the 'generate-and-test' sense, rejecting structures that have already been computed. In [Hendriks 93], the set of proof nets is defined in an alternative, purely inductive way, which does away with proof net conditions for rejecting structures that have already been computed.

DEFINITION 7.7. Proof net conditions ([Roorda 91]). Let t be the term assigned to the $(\cdot)^0$ goal formula.

- (P1) there is precisely one terminal $(\cdot)^0$ formula
- (P2) all axiom substitutions can be performed
- (P3) if t contains a subterm $\lambda x.u$ then x occurs in u and x does not occur outside u
- (P4) every variable assigned to a terminal $(\cdot)^1$ formula occurs in t
- (P5) every subterm of t counts for one in t
- (P6) t has no closed subterms

STRUCTURE LABELING. The semantic labeling of Def 7.6 checks for **LP** derivability, and relies on a geometric criterion (planarity) for the **L** refinement. It is not clear how the geometric approach would generalize to other systems in the categorial landscape, and to the multimodal systems. Below, we give a system of structure labeling, that functions with respect to the 'structural' interpretation of the type language. The labeling regime of Def 7.8 is related to proposals in [Hepple 94, Morrill 95c, Oehrle 95], but makes adjustments for the full multimodal architecture.

DEFINITION 7.8. Structure labels: syntax. The labeling system uses atomic formula labels x and structure labels ${}^{\diamond}\sigma, (\sigma \circ \tau)$, for the $\forall$ formula decomposition nodes. For the $\exists$ nodes, we use *difference* structures: expressions that must be rewritten to structure/formula labels under the residuation reductions of Def 7.9.

σ, τ	$\longrightarrow$	x	(atoms)	$(\sigma \circ \tau)$	(constructor $\bullet$)
		${}^{\diamond}\sigma$	(constructor $\diamond$)	${}^{\triangleleft}(\sigma)$	(left-destructor $\bullet$)
		${}^{\sqcup}\sigma$	(destructor $\diamond$)	$(\sigma)^{\triangleright}$	(right-destructor $\bullet$)
		${}^{\sqcap}\sigma$	(goal $\Box^{\downarrow}$)	$x \backslash \sigma$	(goal $\backslash$)
				σ / x	(goal $/$)

DEFINITION 7.9. Labelled formula decomposition and residuation term reductions (boxed). Positive (antecedent) unfolding $(\cdot)^1$, negative (succedent) unfolding $(\cdot)^0$. We use x, y, z (t, u, v) for object-level formula (structure) labels, Γ, Δ for meta-level structure label variables. Newly introduced formula labels and metavariables in the rules below are chosen fresh.

$$\frac{(t \circ \Delta) : (A)^1 \quad \Delta : (B)^0}{t : (A/B)^1} \qquad \frac{x : (B)^1 \quad \Gamma : (A)^0}{\Gamma / x : (A/B)^0}$$

$$\boxed{(t \circ x)/x \succ t}$$

$$\frac{\Delta : (B)^0 \quad (\Delta \circ t) : (A)^1}{t : (B\backslash A)^1} \qquad \frac{\Gamma : (A)^0 \quad x : (B)^1}{x\backslash\Gamma : (B\backslash A)^0}$$

$$\boxed{x\backslash(x \circ t) \succ t}$$

$$\frac{{}^{\triangleleft}(t) : (A)^1 \quad (t)^{\triangleright} : (B)^1}{t : (A \bullet B)^1} \qquad \frac{\Delta : (B)^0 \quad \Gamma : (A)^0}{(\Gamma \circ \Delta) : (A \bullet B)^0}$$

$$\boxed{({}^{\triangleleft}(t) \circ (t)^{\triangleright}) \succ t}$$

$$\frac{{}^{\sqcup}t : (A)^1}{t : (\Diamond A)^1} \quad \frac{\Gamma : (A)^0}{{}^{\diamond}\Gamma : (\Diamond A)^0} \qquad \frac{{}^{\diamond}t : (A)^1}{t : (\Box^{\downarrow} A)^1} \quad \frac{\Gamma : (A)^0}{{}^{\sqcap}\Gamma : (\Box^{\downarrow} A)^0}$$

$$\boxed{{}^{\diamond\sqcup}t \succ t} \qquad \boxed{{}^{\sqcap\diamond}t \succ t}$$

The basic residuation reductions in Def 7.9 are dictated by the identities for complex formulae $\Diamond A, \Box^{\downarrow} A, A \bullet B, A/B, B\backslash A$. Structural postulates $A \to B$ translate to reductions $\sigma(B) \succ \sigma(A)$, where $\sigma(\cdot)$ is the structure label translation of a formula. The reduction for the distributivity postulate K is given as an illustration in (46). Notice that both residuation reductions and structural postulate reductions are asymmetric, capturing the asymmetry of the derivability relation.

$$\Diamond(A \bullet B) \to \Diamond A \bullet \Diamond B \quad \overset{\sigma(\cdot)}{\rightsquigarrow} \quad ({}^{\diamond}t \circ {}^{\diamond}u) \succ {}^{\diamond}(t \circ u) \tag{46}$$

The parsing problem now assumes the following form. To determine whether a string $x_1 \ldots x_n$ can be assigned the goal type B on the basis of a multiset of lexical assumptions $\Gamma = x_1 : A_1, \ldots, x_n : A_n$, one takes the formula decomposition of $(\Gamma)^1, (B)^0$, and resolves the literals $t : (p)^1$, $\Delta : (p)^0$, with matching $\Delta := t$ under the residuation and/or structural postulate rewritings. The string $x_1 \ldots x_n$ has to be the yield of the structure label assigned to the goal type B.

Example 7.10. Compare the unfoldings (†) for the theorem $A \to \Box^{\downarrow}\Diamond A$ and (‡) for the non-theorem $\Box^{\downarrow}\Diamond A \to A$. Matching $\underline{1}$,$\underline{2}$ gives the instantiation $\Delta := x$. For the goal type $(\Box^{\downarrow}\Diamond A)^0$ we get ${}^{\sqcap\diamond}\Delta = {}^{\sqcap\diamond}x \succ x$. In the case of (‡), matching $\underline{3}$,$\underline{4}$ yields $\Gamma := {}^{\sqcup\diamond}x$ which does not reduce to x.

$$(\dagger) \quad \frac{x : (A)^1 \ \underline{1} \quad \dfrac{\dfrac{\Delta : (A)^0 \ \underline{2}}{{}^{\diamond}\Delta : (\Diamond A)^0}}{{}^{\sqcap\diamond}\Delta : (\Box^{\downarrow}\Diamond A)^0}$$

$$(\ddagger) \quad \frac{\dfrac{\dfrac{{}^{\sqcup\diamond}x : (A)^1 \ \underline{3}}{{}^{\diamond}x : (\Diamond A)^1}}{x : (\Box^{\downarrow}\Diamond A)^1} \quad \Gamma : (A)^0 \ \underline{4}}$$

EFFICIENT COMPILATION TECHNIQUES. The labeling regime of Def 7.9 covers the multimodal architecture in a general fashion. In designing efficient compilation techniques one can exploit the properties of specific multimodal grammars. Such techniques have been developed in [Morrill 95b, Morrill 95c] for the grammar fragment covered by [Morrill 94a] and some extensions. With a restriction on the use of the product connective, the compiler can translate lexical type assignments into a higher-order *clausal* fragment of linear logic. The clauses are processed by a linear-logical clausal engine and the normalised structure labels and semantic labels yielding the string are enumerated. The structural, sublinear properties, are represented by the term structure of the linear clauses. Satisfaction of constraints

under associativity can be met not by enumerating and testing unifiers, but by propagating successive constraints through string position or difference list representations, as used in the compilation of context-free grammar. One thus obtains a linear logical programming paradigm for categorial grammar which is the counterpart of the logic programing paradigm for phrase structure grammar. But whereas the latter lacks resource-consciousness (in a derivation a rule may be used once, more than one, or not at all), sensitivity for the lexical resources processed is a built-in feature of the categorial paradigm.

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 **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 Categorial 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 Categorial 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 Categorial 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. Skordev (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 Categorial Unification Grammar*. Ph.D. Dissertation, Groningen.

[Bouma 88b] Bouma, G. (1988), 'Modifiers and specifiers in Categorial 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 Categorial 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, Poznán. 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 Categorial 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 categorical 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, Debrečen*, 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.