# Analyzing the Core of Categorial Grammar

Carlos Areces
ILLC, University of Amsterdam
carlos@science.uva.nl

Raffaella Bernardi
UiL-OTS, University of Utrecht
Raffaella.Bernardi@let.uu.nl

**Abstract**

Even though residuation is at the core of Categorial Grammar [11], it is not always immediate to realize how standard logic systems like Multi-modal Categorial Type Logics (MCTL) [17] actually *embody* this property. In this paper we focus on the basic system NL [12] and its extension with unary modalities NL($\diamond$) [16], and we spell things out by means of Display Calculi (DC) [3, 10]. The use of structural operators in DC permits a sharp distinction between the core properties we want to impose on the logic system and the way these properties are projected into the logic operators. We will show how we can obtain Lambek residuated triple $\backslash$, $/$ and $\bullet$ of binary operators, and how the operators $\diamond$ and $\Box^{\downarrow}$ introduced by Moortgat in [16] are indeed their unary counterpart.

In the second part of the paper we turn to other important algebraic properties which are usually investigated in conjunction with residuation [5]: Galois and dual Galois connections. Again, DC let us readily define logic calculi capturing them, and we will discuss different possibilities in which the logic operators so obtained can interact with those obtained from residuation. We also provide preliminary ideas on how to use them when modeling linguistic phenomena.

## 1 Categorial Grammar

In the Categorial Grammar approach to natural language analysis [1, 2] sentences are seen as sequences of function applications starting from the categories assigned to the lexical items in the lexicon. In [11] Lambek shows that the categorial grammar intuitions can be formalized into a logic calculus: the grammaticality of a sentence can be decided by means of logic rules, if we consider categories as logic formulas. This idea is at the heart of what are today called Categorial Type Logics (CTL's). By means of these logics we can investigate logic properties of linguistic composition (like the impact of associativity and permutation in natural language phenomena). In other words, CTL's can account for grammaticality in a purely proof-theoretic way, and moreover, typical CTL's are *decidable*, that is they are amenable to computational treatment. More precisely, the standard categorial approach is to develop the correct "type assignment" for basic lexical items, from which certain linguistic phenomena will be predicted by the logic. Once the basic types have been fixed, parsing a linguistic expression to check its membership to a given type, amounts to inferring the type in the logic system from the types assigned to its lexical components. Automatic systems like Grail [18] have been specially devised for this task, being able to handle different CTL's.

In addition to this pure proof-theoretic use of categorial systems, there is also an important semantic byproduct: the interpretation of (formally, the lambda term associated to) a linguistic expression can be obtained while inferring its type [17]. This connection offers a rich framework where linguistic issues can be investigated from all three different points of view: purely syntactic checking of *type composition*, the *compositional meaning* of the linguistic expression, and their interface. Actually, the CTL approach can be understood as the result of analysing this three sided linguistic picture (of syntax, semantics and their interface) from the standpoint of its syntactic vertex.

Types, how to form them and which inference patterns they give rise to, are core issues in categorial type logics. In this paper we will study some of these systems, analyse how their inference rules have been obtained, and in which way they define the behavior of their type forming operators.

The original sequent system NL introduced by Lambek in [12] consisted of the operators $\backslash$, $/$ and $\bullet$, indirectly governed by the algebraic law of residuation. While $\bullet$ can be seen as playing the role of linguistic composition, $\backslash$ and $/$ allow the definition of functional types, which are sensitive to order. In Section 2.1 we will show how display calculi (DC) [3, 10] let us *directly* specify the residuation law by means of structural rules. The residuation behavior is then projected into the logic operators in a clean way.

Modern systems like those discussed in [17] are richer than NL, incorporating unary logic operators and special devices to handle structures. In Section 2.2, we study NL($\Diamond$) introduced by Moortgat in [16] by extending NL with unary operators $\Diamond$ and $\Box^{\downarrow}$, and show that they are indeed the unary counterpart of Lambek's residuated triple. This is no new result, the standard proof proceeds by showing that derivations in NL($\Diamond$) exactly match theorems obtained from the algebraic laws governing (unary and binary) residuation (see [16]). But this method provides little help on how to *actually obtain* the sequent rules of $\mathsf{NL}(\Diamond)$ —capturing residuation and no more— in the first place; $\mathsf{DC}$ instead achieve this straightforwardly.

Interestingly, $\mathsf{DC}$ do not let us handle only residuation. The method described in Section 2 can be applied to other algebraic properties. In Section 3 we investigate Galois and dual Galois connections, and define logic calculi capturing them. Galois connections are interesting because they give rise to new derivability relations among types while their composition is still a closure operator (as we discuss in Section 5). In Section 4 we discuss different possibilities in which the logic operators governed by the Galois and dual Galois laws can interact with those obtained from residuation.

We believe that the main contribution of this paper is methodological, and that display calculi do provide new light and help understand standard categorial type logics. More generally, we describe a recipe that help us explore a landscape of algebraic principles. We exemplify this method by showing how to extend the basic calculus $\mathsf{NL}$ with residuated, Galois and dual Galois connected operators of different arities in a systematic way.

## 2 Capturing Residuation

We start by formally introducing residuation and its ramifications in modern categorial type logics (see [16] for an excellent and much more complete discussion).

The property of residuation arises in the study of order-preserving mappings [8, 6]. Intuitively, in any partial order with a "product-like" operator $\bullet$, the (right) *residual* for an element $a$ with respect to $b$ is the largest $c$ such that $b \bullet c$ is less than or equal to $a$. In case the residual always exists for any two elements $a, b$ in the structure, we can define the function $\cdot\backslash\cdot$ returning it. $\backslash$ and $\bullet$ are said to be *residuated.* If $\bullet$ is not permutative, then also a notion of left-residual naturally arises.

For example, in the rational numbers (without 0), given any rational $a$, the residual with respect to $b$ is simply $\frac{a}{b}$, and in this structure product and division are residuated functions.

More abstractly, the notion of residuated functions can be generally introduced for maps with $n$-ary arguments, but we restrict ourselves to unary and binary functions.

**Definition 2.1 (Residuation)** Let $\mathcal{A}_i = (A_i, \sqsubseteq_{A_i})$ be a partially ordered set. A pair of functions $(f, g)$ such that $f : A_1 \to A_2$ and $g : A_2 \to A_1$ forms a *residuated pair* if [RES$_1$] holds.

$$[\mathrm{RES}_1] \qquad \forall x \in A_1, y \in A_2 \begin{pmatrix} fx & \sqsubseteq_{A_2} & y & \text{iff} \\ x & \sqsubseteq_{A_1} & gy & \end{pmatrix}.$$

A triple of functions $(f, g, h)$ such that $f : A_1 \times A_2 \to A_3$, $g : A_1 \times A_3 \to A_2$, $h : A_3 \times A_2 \to A_1$ forms a *residuated triple* if [RES$_2$] holds.

$$[\mathrm{RES}_2] \qquad \forall x \in A_1, y \in A_2, z \in A_3 \begin{pmatrix} f(x,y) & \sqsubseteq_{A_3} & z & \text{iff} \\ y & \sqsubseteq_{A_2} & g(x,z) & \text{iff} \\ x & \sqsubseteq_{A_1} & h(z,y) & \end{pmatrix}.$$

In both cases the function $f$ is said to be the *head* of the residuated pair or triple.

**Remark 2.2** It is important to mention that residuation has an impact on monotonicity behavior. In fact, saying that $(f, g)$ is a residuated pair is equivalent to the conditions *i*) and *ii*) below, where we write $f$ is a $[\uparrow]$-function ($f$ is a $[\downarrow]$-function) meaning that $f$ is upwards (downwards) monotonic in its argument,

*i.* $f$ and $g$ are $[\uparrow]$-functions.
*ii.* $\forall y \in A_2 (fgy \sqsubseteq_{A_2} y)$ and $\forall x \in A_1 (x \sqsubseteq_{A_1} gfx)$.

Similarly, saying that $(f, g, h)$ is a residuated triple is equivalent to requiring

*i.* $f$ is a $[\uparrow, \uparrow]$-function, $g$ is an $[\downarrow, \uparrow]$-function and $h$ is an $[\uparrow, \downarrow]$-function.
*ii.* $\forall x \in A_1, y \in A_2, z \in A_3 ((f(x, g(x, z)) \sqsubseteq_{A_3} z) \ \& \ (y \sqsubseteq_{A_2} g(x, f(x, y)))$ $\& \ (f(h(z, y), y) \sqsubseteq_{A_3} z) \ \& \ (x \sqsubseteq_{A_1} h(f(x, y), y)))$.

In what follows we will be mainly interested in logic operations $O_i : \mathsf{FORM} \to \mathsf{FORM}$, and we will investigate their behavior with respect to the poset $\langle \mathsf{FORM}, \vdash \rangle$ where $\vdash$ is the derivability relation.

Categorial type logics are also known as "logics of residuation." Let us see why, by considering the system NL introduced in [12].

**Definition 2.3 (Logic language of NL)** Given the set ATOM of atomic propositional symbols, the logic language of NL is defined recursively as

$$\mathsf{FORM} ::= \mathsf{ATOM} \mid \mathsf{FORM}/\mathsf{FORM} \mid \mathsf{FORM}\backslash\mathsf{FORM} \mid \mathsf{FORM} \bullet \mathsf{FORM}.$$

A Hilbert style deductive system for NL is given as follows.

**Definition 2.4 (NL: Hilbert system)** The system NL, is defined by the axioms below. Given $A, B, C \in \mathsf{FORM}$

$$\begin{array}{rl}
[\mathrm{REFL}] & \vdash A \longrightarrow A, \\
[\mathrm{TRANS}] & \text{If} \vdash A \longrightarrow B \text{ and} \vdash B \longrightarrow C, \text{ then} \vdash A \longrightarrow C, \\
[\mathrm{RES}] & \vdash A \longrightarrow C/B \text{ iff} \vdash A \bullet B \longrightarrow C \text{ iff} \vdash B \longrightarrow A\backslash C.
\end{array}$$

NL is commonly called the pure logic of residuation, and rightly so as we can see from its axiomatic presentation. [REFL] and [TRANS] define minimal properties for the inference relation $\longrightarrow$ while [RES] characterizes $\bullet$, $\backslash$ and $/$ as a residuated triple[1].

The Hilbert style presentation of NL clearly shows that residuation directly governs the behavior of its type forming operators. But even though Hilbert style deductive system can be effectively used when establishing model-theoretic properties like completeness, it is not appropriate for proof-theoretic investigations. In particular, the [TRANS] rule above violates the sub-formula property, introducing non determinism in the proof search.

As with classical propositional logic, an alternative is the formulation of an equivalent Gentzen style presentation, in which the use of the counterpart of [TRANS], the [Cut] rule, can be proved to be redundant ([Cut] elimination). Standard Gentzen systems in which [Cut] can be eliminated, do enjoy the sub-formula property, and hence the search space for proofs of a given sequent is finite. This good computational behavior makes these systems well suited to the study of the inference relations between types.

While in the axiomatic presentation the derivability relation holds between formulas of the logic language, in a Gentzen system it is stated in terms of *sequents*: pairs $\Gamma \vdash A$ where $\Gamma$ is a structured configuration of formulas or *structural term* and $A$ is a logic formula. The set TERM of structural terms needed for a sequent presentation of NL is very simple.

$$\mathsf{TERM} ::= \mathsf{FORM} \mid (\mathsf{TERM}, \mathsf{TERM}).$$

The logic rules in the Gentzen system for NL are given in Figure 1 below. In the figure, $A, B, C$ are formulas, $\Gamma, \Delta$ are structural terms and the notation $\Gamma[\varphi]$ is used to single out a particular instance of the substructure $\varphi$ in $\Gamma$.

As we can see from inspecting these rules, it is not immediately obvious that they are characterizing the same derivability relation as the one characterized by the Hilbert presentation of NL. To establish the equivalence

[1] [RES] could also be understood as a kind of deduction theorem. But while a deduction theorem is better seen as a link between the meta-language and the object language, [RES] relates three operators in the object language.

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

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

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

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

Figure 1: Gentzen sequent calculus for NL

between the two presentations, define the translation $\cdot^t : \mathsf{TERM} \to \mathsf{FORM}$ as

$$\begin{aligned} (\Gamma_1, \Gamma_2)^t &= (\Gamma_1^t \bullet \Gamma_2^t), \\ A^t &= A, \text{ for } A \in \mathsf{FORM}. \end{aligned}$$

**Proposition 2.5 (See [11, 12])** *If $\vdash A \longrightarrow B$ is a theorem of the Hilbert axiomatization of NL then there is a Gentzen proof of $A \vdash B$. And for every proof of a sequent $\Gamma \vdash B$, $\vdash \Gamma^t \longrightarrow B$ is a theorem.*

The system presented in Figure 1 includes the [Cut] rule but Lambek proved, also in [11], that the rule is admissible, in the sense that it does not increase the set of theorems that can already be derived using just the other rules.

**Proposition 2.6 ([Cut] elimination and decidability)** *The [Cut] rule is admissible in NL, and hence the system is decidable.*

There seems to be a tension in the standard approach we described above. On the one hand, while the Hilbert calculus crisply captures the notion of residuation we are interested in, it is not appropriate for proof-theoretic manipulation. On the other hand, the decidable [Cut] free sequent presentation hides the residuated behavior of the operators, requiring a "verification check" as shown in Proposition 2.5. In the next section we will explain how display calculi are able to resolve this tension.

## 2.1 Displaying Residuation

Display calculus, introduced by Belnap in [3], is a general Gentzen style proof-theoretic framework designed to capture many different logics in one uniform setting. DC generalize Gentzen's notion of structures using multiple, complex, structural connectives. One of the main characteristics of DC is a general cut-elimination theorem, which applies whenever the rules of the display calculus obey certain, easily checked, conditions.

We will base our presentation on the system introduced by Goré in [10]. The main innovation of Goré's system over Belnap's concerns the use of additional structural connectives to capture the inherent duality of every logic, by means of dual sets of display postulates. Building on these features, DC obtain the fundamental property which gives them their name: any particular constituent of a sequent can be turned into the whole of the right or left side by moving other constituents to the other side. This property is strongly used in the general cut-elimination method. But more interesting than the display property for our approach, is the ability of DC to define the behavior of their logic operators in terms of *structural properties* — sequent rules involving only structural operators.

Let's start by introducing the appropriate logic and structural language for the DC we want to investigate.

**Definition 2.7 (DC language)** Given a set ATOM of atomic propositional symbols and the sets $\mathsf{OP}_s = \{;, <, >\}$ and $\mathsf{OP}_l = \{\otimes, \leftarrow, \rightarrow\}$ of structural and logic operators respectively, the set FORM of logic formulas and the set STRUCT of structural formulas are defined as

$$\begin{array}{ll} \mathsf{FORM} ::= & \mathsf{ATOM} \mid \mathsf{FORM} \otimes \mathsf{FORM} \mid \\ & \mathsf{FORM} \leftarrow \mathsf{FORM} \mid \mathsf{FORM} \rightarrow \mathsf{FORM}. \\ \mathsf{STRUCT} ::= & \mathsf{FORM} \mid \mathsf{STRUCT}; \mathsf{STRUCT} \mid \\ & \mathsf{STRUCT} < \mathsf{STRUCT} \mid \mathsf{STRUCT} > \mathsf{STRUCT}. \end{array}$$

The behavior of the structural operators is explicitly expressed by means of display postulates. In what follows we will use variables $X, Y, Z, V, W$ to denote structural formulas, and reserve $A, B, C$ for logic formulas. In the case of residuation, we can directly express that $(;, <, >)$ is a residuated triple by the following structural rule [rp]

$$\begin{array}{c} Y \vdash X > Z \\ \hline\hline X; Y \vdash Z \\ \hline\hline X \vdash Z < Y \end{array}$$

What remains now is to project the residuation behavior of $(;, <, >)$ into the corresponding logic operators $(\otimes, \leftarrow, \rightarrow)$. The general methodology is described in detail in [9]. In a nutshell, it works as follows. We are in search of a right and left introduction rule for each of the logic operators, we can obtain [$\otimes\vdash$], [$\vdash\leftarrow$] and [$\vdash\rightarrow$] directly from [rp] by projection. In the literature on display calculi these rules are usually called *rewrite rules* (see Figure 2).

To obtain the still missing rules we have to work only slightly harder. As we pointed out in Remark 2.2, from the fact that $(;,<,>)$ are residuated we know their monotonicity behavior, and this is exactly what we need.

Let $s$ be a structural operator and $l$ its corresponding logic counterpart. In the schema below we will select whether the consequent of the rule is $s(X,Y)\vdash l(V,D)$ or $l(X,Y)\vdash s(V,W)$ depending on the needed rule.

$$\frac{V\vdash X \quad W\vdash Y}{[l,s](X,Y)\vdash[s,l](V,W)}\ \text{if } s \text{ is } [\downarrow,\downarrow] \qquad \frac{X\vdash V \quad Y\vdash W}{[l,s](X,Y)\vdash[s,l](V,W)}\ \text{if } s \text{ is } [\uparrow,\uparrow]$$

$$\frac{X\vdash V \quad W\vdash Y}{[l,s](X,Y)\vdash[s,l](V,W)}\ \text{if } s \text{ is } [\uparrow,\downarrow] \qquad \frac{V\vdash X \quad Y\vdash W}{[l,s](X,Y)\vdash[s,l](V,W)}\ \text{if } s \text{ is } [\downarrow,\uparrow]$$

Applying the schema above, we obtain [$\vdash\otimes$], [$\leftarrow\vdash$], and [$\rightarrow\vdash$]. The full set of rules is shown in Figure 2.

*Logic Rules*

$$\frac{A\vdash X \quad Y\vdash B}{A\leftarrow B\vdash X<Y}\,[\leftarrow\vdash] \qquad \frac{Z\vdash A<B}{Z\vdash A\leftarrow B}\,[\vdash\leftarrow]$$

$$\frac{A;B\vdash Z}{A\otimes B\vdash Z}\,[\otimes\vdash] \qquad \frac{Y\vdash B \quad X\vdash A}{Y;X\vdash B\otimes A}\,[\vdash\otimes]$$

$$\frac{X\vdash A \quad B\vdash Y}{A\rightarrow B\vdash X>Y}\,[\rightarrow\vdash] \qquad \frac{Z\vdash A>B}{Z\vdash A\rightarrow B}\,[\vdash\rightarrow]$$

Figure 2: Logic rules for residuated binary operators

The rules will immediately encode the proper tonicity of the operator. It is also easy to prove that the logic operators indeed satisfy the residuation property. We show two of the required four derivations below.

$$\frac{\dfrac{\dfrac{B\vdash A\rightarrow C \quad \dfrac{A\vdash A \quad C\vdash C}{A\rightarrow C\vdash A>C}\,[\rightarrow\vdash]}{B\vdash A>C}\,[\text{Cut}]}{A;B\vdash C}\,[\text{rp}]}{A\otimes B\vdash C}\,[\otimes\vdash] \qquad \frac{\dfrac{\dfrac{\dfrac{A\vdash A \quad B\vdash B}{A;B\vdash A\otimes B}\,[\vdash\otimes] \quad A\otimes B\vdash C}{A;B\vdash C}\,[\text{Cut}]}{B\vdash A>C}\,[\text{rp}]}{B\vdash A\rightarrow C}\,[\vdash\rightarrow]$$

And in a similar way we can prove the "composition property" we mentioned in Remark 2.2.

As we can see, DC provide guidance in our logic engineering task of designing a sequent calculus characterizing the behavior of a triple of residuated operators. Moreover, we can readily verify the conditions specified by Belnap and conclude that the calculus is cut-free.

If we compare the calculus just obtained with the one introduced in Figure 1 we immediately notice similarities, but also important differences, the most relevant being the presence of only one structural operator, and the restriction to a single formula in the right hand side of sequents. It is not too difficult to restrict the language to obtain a perfect match (but of course, in doing so we would be relinquishing the display property, and "abandoning" DC and its general theorem concerning cut-elimination). Consider, for example, the [$\rightarrow\vdash$] rule

$$\frac{X \vdash A \quad B \vdash C}{A \rightarrow B \vdash X > C}\,[\rightarrow\vdash] \quad \text{by [rp]} \quad \cfrac{\cfrac{X \vdash A \quad B \vdash C}{A \rightarrow B \vdash X > C}\,[\rightarrow\vdash]}{X ; A \rightarrow B \vdash C}\,[\mathrm{rp}] \quad \text{hence} \quad \frac{X \vdash A \quad B \vdash C}{X ; A \rightarrow B \vdash C}\,.$$

By replacing ; by , and $\rightarrow$ by $\backslash$ and adding structural contexts (which are now required given that we have lost the display property) we obtain [$\backslash$L]

$$\frac{X \vdash A \quad \Gamma[B] \vdash C}{\Gamma[X, A\backslash B] \vdash C}\,[\backslash \mathrm{L}].$$

In the next section we will treat in detail unary residuation and investigate the pair of operators $\diamond$ and $\Box^{\downarrow}$.

## 2.2 The Unary Operators

The system NL($\diamond$) introduced in [16, 17] is obtained from NL by the addition of the unary modalities $\diamond$ and $\Box^{\downarrow}$, and it is actively used in the analysis of linguistic phenomena. But $\diamond$ and $\Box^{\downarrow}$ have been sometimes "looked down upon" as extraneous to a calculus of pure residuation.

We will show that we can straightforwardly mimic what we did in the previous section. Starting by spelling out the law of residuation for unary functions, we derive a sequent calculus that can be compiled into the standard calculus for NL($\diamond$). Let's start by defining the proper logic and structural languages[2].

[2] Again we will start by following the notation of Goré in [10] to revert to the standard notation used in NL($\diamond$) during the compilation step.

**Definition 2.8 (Logic and structural languages for a DC presentation of $\mathsf{NL}(\Diamond)$)** Given a set $\mathsf{ATOM}$ of atomic propositional symbols and the sets $\mathsf{OP}_s = \{\bullet, \circ\}$ and $\mathsf{OP}_l = \{\blacklozenge, \Box\}$ of structural and logic operators, the set $\mathsf{FORM}$ of logic formulas and the set $\mathsf{STRUCT}$ of structural formulas for a display calculus presentation of $\mathsf{NL}(\Diamond)$ are defined as

$$\mathsf{FORM} ::= \mathsf{ATOM} \mid \blacklozenge\mathsf{FORM} \mid \Box\mathsf{FORM}.$$

$$\mathsf{STRUCT} ::= \mathsf{FORM} \mid \bullet\,\mathsf{STRUCT} \mid \circ\,\mathsf{STRUCT}.$$

Again we start by specifying the residuated behavior of the structural pair $(\circ, \bullet)$,

$$\frac{\bullet X \vdash Y}{X \vdash \circ Y}\ [\mathrm{rp}].$$

And we obtain the rules for the logic operators by projection and monotonicity behavior. The full set of rules is given in Figure 3 below.

*Logic Rules*

$$\frac{A \vdash X}{\Box A \vdash \circ X}\ [\Box\vdash] \qquad \frac{X \vdash \circ A}{X \vdash \Box A}\ [\vdash\Box]$$

$$\frac{\bullet A \vdash X}{\blacklozenge A \vdash X}\ [\blacklozenge\vdash] \qquad \frac{X \vdash A}{\bullet X \vdash \blacklozenge A}\ [\vdash\blacklozenge]$$

Figure 3: Logic rules for residuated unary operators

We can prove that $(\blacklozenge, \Box)$ is a residuated pair.

$$\dfrac{\dfrac{\dfrac{A \vdash \Box B \quad \dfrac{B \vdash B}{\Box B \vdash \circ B}\,[\Box\vdash]}{A \vdash \circ B}\,[\mathrm{Cut}]}{\bullet A \vdash B}\,[\mathrm{rp}]}{\blacklozenge A \vdash B}\,[\blacklozenge\vdash] \qquad \dfrac{\dfrac{\dfrac{\dfrac{A \vdash A}{\bullet A \vdash \blacklozenge A}\,[\vdash\blacklozenge] \quad \blacklozenge A \vdash B}{\bullet A \vdash B}\,[\mathrm{Cut}]}{A \vdash \circ B}\,[\mathrm{rp}]}{A \vdash \Box B}\,[\vdash\Box]$$

Now we "compile" the structural postulate [rp] to obtain the logic rules in the standard Gentzen presentation of $\mathsf{NL}(\Diamond)$, as we did in the case of binary operators. We spell out the needed steps for the $\Box$ operator and obtain the rules $[\Box^{\downarrow}\mathrm{L}]$ and $[\Box^{\downarrow}\mathrm{R}]$ as presented in [17].

$$\dfrac{A \vdash B}{\Box A \vdash \circ B}\,[\Box \vdash] \quad \text{by [rp]} \quad \dfrac{\dfrac{A \vdash B}{\Box A \vdash \circ B}\,[\Box \vdash]}{\bullet \Box A \vdash B}\,[\text{rp}] \quad \text{by compilation} \quad \dfrac{\Gamma[A] \vdash B}{\Gamma[\langle \Box^{\downarrow} A \rangle] \vdash B}\,[\Box^{\downarrow}\text{L}].$$

$$\dfrac{X \vdash \circ A}{X \vdash \Box A}\,[\Box \vdash] \quad \text{by [rp]} \quad \dfrac{\dfrac{\bullet X \vdash A}{X \vdash \circ A}\,[\text{rp}]}{X \vdash \Box A}\,[\Box \vdash] \quad \text{by compilation} \quad \dfrac{\langle X \rangle \vdash A}{X \vdash \Box^{\downarrow} A}\,[\Box^{\downarrow}\text{R}].$$

Another way to try to understand the sequent rules for the unary residuated operators is by starting from the binary residuated triple $(\bullet, /, \backslash)$ and "unarize" them by fixing one of the parameters. We think that the path we followed above is clearer.

As we said in the introduction, DC are not limited to residuation properties (even though they are an important example, as residuation aids in achieving the display property). The method we have used above can handle other kind of algebraic properties, assuming that they can be encoded in terms of structural rules. In the next section we turn to Galois and dual Galois connections.

# 3 Capturing Galois and Dual Galois Connections

If we look back at NL and at examples of how it is used in modeling linguistic phenomena, we notice that sometimes only the $\backslash$ and $/$ operators are required, and their behavior is not characterized by a residuation law. Actually, $\backslash$ and $/$ form a Galois connection when their positive argument is fixed. This is exactly what is used in CTL, for example, to account for the rising of noun phrases $np$ to general quantifiers $(s/np)\backslash s$. The fact that $/$ and $\backslash$ are Galois connected means exactly that for any two types $A$ and $B$ we can infer $(B/A)\backslash B$ from $A$.

Under this light, even though calling NL the pure calculus of residuation is correct, it is also misleading as it hides the fact that a Galois connected pair is also "living" inside as a subsystem. And in line with the work we did in Section 2.2 we could wonder whether we can also extend NL($\Diamond$) with a pair of independent unary Galois connected operators. But let's start by formally introducing the algebraic properties we want to investigate.

**Definition 3.1 (Galois connections)** Let $\mathcal{A}_i = (A_i, \sqsubseteq_{A_i})$ be a partially ordered set. Consider a pair of functions $f : A_1 \to A_2$ and $g : A_2 \to A_1$. The pair $(f, g)$ is called a *Galois connection* if [GC] below holds.

$$[\text{GC}] \qquad \forall x \in A_1, y \in A_2 \left( \begin{array}{llll} y & \sqsubseteq_{A_2} & fx & \text{iff} \\ x & \sqsubseteq_{A_1} & gy & \end{array} \right).$$

The pair $(f, g)$ is called a *dual Galois connection* if [DGC] below holds.

$$\text{[DGC]} \qquad \forall x \in A_1, y \in A_2 \left( \begin{array}{cccc} fx & \sqsubseteq_{A_2} & y & \text{iff} \\ gy & \sqsubseteq_{A_1} & x & \end{array} \right).$$

As with residuation, there is an equivalent formulation of these properties in terms of their monotonicity behavior and a composition rule. [GC] for example, is equivalent to require that $f$ and $g$ are both [$\downarrow$]-functions, and that for all $x$, $x \sqsubseteq fgx$, and $x \sqsubseteq gfx$ (here again, we just consider $f$ and $g$ as functions defined on the same poset).

The steps we will take to provide a $\mathsf{DC}$ calculus encoding [GC] and [DGC] should be by now well known. We will only provide details for [GC]. We start by explicitly writing the algebraic property characterizing a Galois connection for a pair of structural operators $(\sharp, \flat)$.

$$\frac{Y \vdash \flat X}{X \vdash \sharp Y}\ \text{[gc]}.$$

We now project this behavior into the logic operators $({}^0(\cdot), (\cdot)^0)$ as it is shown in Figure 4.

| *Logic Rules* | |
|---|---|
| $\dfrac{Z \vdash A}{{}^0(A) \vdash \flat Z}\ [{}^0(\cdot) \vdash]$ | $\dfrac{Z \vdash \flat A}{Z \vdash {}^0(A)}\ [\vdash {}^0(\cdot)]$ |
| $\dfrac{Z \vdash A}{(A)^0 \vdash \sharp Z}\ [(\cdot)^0 \vdash]$ | $\dfrac{Z \vdash \sharp A}{Z \vdash (A)^0}\ [\vdash (\cdot)^0]$ |

Figure 4: Logic rules for Galois connected unary operators

To move closer to standard sequent presentations of $\mathsf{CTL}$, we need to compile [gc] into the logic rules. We can take $[{}^0(\cdot) \vdash]$ and $[(\cdot)^0 \vdash]$ as they are as our $[{}^0(\cdot)\mathrm{L}]$ and $[(\cdot)^0\mathrm{L}]$. To obtain $[{}^0(\cdot)\mathrm{R}]$ and $[(\cdot)^0\mathrm{R}]$ we need to apply [gc],

$$\frac{Z \vdash \flat A}{Z \vdash {}^0(A)}\ [\vdash {}^0(\cdot)] \quad \text{by [gc]} \quad \dfrac{\dfrac{A \vdash \sharp Z}{Z \vdash \flat A}\ \text{[gc]}}{Z \vdash {}^0(A)}\ [\vdash {}^0(\cdot)] \quad \text{by compilation} \quad \frac{A \vdash \sharp Z}{Z \vdash {}^0(A)}\ [{}^0(\cdot)\mathrm{R}].$$

$$\frac{Z \vdash \sharp A}{Z \vdash (A)^0}\ [\vdash (\cdot)^0] \quad \text{by [gc]} \quad \dfrac{\dfrac{A \vdash \flat Z}{Z \vdash \sharp A}}{Z \vdash (A)^0}\ [\vdash (\cdot)^0] \quad \text{by compilation} \quad \frac{A \vdash \flat Z}{Z \vdash (A)^0}\ [(\cdot)^0\mathrm{R}].$$

The full set of rules obtained is shown in Figure 5. Notice that given the nature of Galois connections (which involves a permutation in the order of the poset), it is not possible to eliminate the structural operators from the right hand side of the sequents. This is an important difference with respect to what we obtained in the previous sections.

*Logic Rules*

$$\frac{Z \vdash A}{{}^0(A) \vdash \flat Z}\ [{}^0(\cdot)\mathrm{L}] \qquad \frac{A \vdash \sharp Z}{Z \vdash {}^0(A)}\ [{}^0(\cdot)\mathrm{R}]$$

$$\frac{Z \vdash A}{(A)^0 \vdash \sharp Z}\ [(\cdot)^0\mathrm{L}] \qquad \frac{A \vdash \flat Z}{Z \vdash (A)^0}\ [(\cdot)^0\mathrm{R}]$$

$$\frac{X \vdash Y \quad Y \vdash Z}{X \vdash Z}\ [\mathrm{Cut}]$$

Figure 5: Compiled logic rules for Galois connected unary operators

The proofs below show that the $(\cdot)^0$ and ${}^0(\cdot)$ operators are indeed Galois connected,

$$\cfrac{\cfrac{A \vdash (B)^0 \quad \cfrac{B \vdash B}{(B)^0 \vdash \sharp B}\ [(\cdot)^0\mathrm{L}]}{A \vdash \sharp B}\ [\mathrm{Cut}]}{B \vdash {}^0(A)}\ [{}^0(\cdot)\mathrm{R}] \qquad \cfrac{\cfrac{A \vdash {}^0(B) \quad \cfrac{B \vdash B}{{}^0(B) \vdash \flat B}\ [{}^0(\cdot)\mathrm{L}]}{A \vdash \flat B}\ [\mathrm{Cut}]}{B \vdash (A)^0}\ [(\cdot)^0\mathrm{R}]$$

That is, the rule [gc] holds for ${}^0(\cdot)$ and $(\cdot)^0$. Moreover, the operators satisfy the appropriate Galois composition laws [gcl].

$$\cfrac{\cfrac{A \vdash A}{(A)^0 \vdash \sharp A}\ [(\cdot)^0\mathrm{L}]}{A \vdash {}^0((A)^0)}\ [{}^0(\cdot)\mathrm{R}] \qquad \cfrac{\cfrac{A \vdash A}{{}^0(A) \vdash \flat A}\ [{}^0(\cdot)\mathrm{L}]}{A \vdash ({}^0(A))^0}\ [(\cdot)^0\mathrm{R}]$$

¿From these, the fact that the operators are [$\downarrow$]-functions follows immediately.

$$\cfrac{\cfrac{A \vdash B}{A \vdash {}^0((B)^0)}\ [\mathrm{gcl}]}{(B)^0 \vdash (A)^0}\ [\mathrm{gc}] \qquad \cfrac{\cfrac{A \vdash B}{A \vdash ({}^0(B))^0}\ [\mathrm{gcl}]}{{}^0(B) \vdash {}^0(A)}\ [\mathrm{gc}]$$

## 4 Relating Galois and Residuation

In the previous section we have analyzed how residuated and Galois connected pairs and triples lived at the core of categorial type logics, and shown how display calculi provides an interesting methodology to handle them. In this section we will digress into a different but strongly related topic. We will discuss different ways in which we can build further interaction among the operators, by analyzing alternative semantics (see Dunn's work on Gaggle Theory [7] for a discussion of semantics for residuation and (dual) Galois connections, and [9] for a general presentation in the framework of display calculi).

By adding the rules in Figure 5 to $\mathsf{NL}(\diamond)$ we obtain a system with both residuated and Galois connected unary operators that we will call $\mathsf{NL}(\diamond, {}^0(\cdot))$, but the operators are totally independent and the interaction among them is minimal. We can make this precise by analyzing a possible semantics for these operators.

**Definition 4.1 (Kripke models and satisfiability relation)** A *Kripke model* is a tuple $\mathcal{M} = \langle W, R_1, R_2 \rangle$ where $W$ is a non empty set and $R_1$ and $R_2$ are binary accessibility relations on $W$. A *satisfiability relation* $\models$ defining the behavior of the $\diamond, \Box^{\downarrow}, {}^0(\cdot)$ and $(\cdot)^0$ operators can be defined as follows. Given $\mathcal{M}$ a Kripke model, and $x, y \in W$,

$$\begin{array}{rcl}
\mathcal{M}, x \models \diamond A & \text{iff} & \exists y.(R_1xy \ \& \ \mathcal{M}, y \models A) \\
\mathcal{M}, y \models \Box^{\downarrow} A & \text{iff} & \forall x.(R_1xy \Rightarrow \mathcal{M}, x \models A) \\
\mathcal{M}, x \models {}^0(A) & \text{iff} & \forall y.(R_2xy \Rightarrow \mathcal{M}, y \not\models A) \\
\mathcal{M}, y \models (A)^0 & \text{iff} & \forall x.(R_2xy \Rightarrow \mathcal{M}, x \not\models A).
\end{array}$$

Proving that so defined, $(\diamond, \Box^{\downarrow})$ is a unary residuated pair and that $({}^0(\cdot), (\cdot)^0)$ are Galois connected is straightforward. It is also not too difficult to prove that $\mathsf{NL}(\diamond, {}^0(\cdot))$ is complete for this semantics. And as we can see by inspecting our definition of Kripke models, the accessibility relations $R_1$ and $R_2$ governing the residuated and Galois connected operators, respectively, are totally independent of each other.

If we want to create further interaction among the operators we can consider different ways of relating $R_1$ and $R_2$. A natural, but perhaps too strong, option is to set $R_1 = R_2$, and let the operators be defined by a unique binary accessibility relation (such a condition will be accounted for on the proof-theoretic side by the addition of further structural rules to $\mathsf{NL}(\diamond, {}^0(\cdot))$). Under this assumption, some interesting properties can be proved. For example we can show by means of a semantic argument that $(\Diamond A)^0 \longrightarrow \Box^{\downarrow}({}^0(A))$ holds as follows. Given a model $\mathcal{M}$ and $x$ in $W$, we assume $\mathcal{M}, x \models (\Diamond A)^0$ and, for contradiction, suppose that $\mathcal{M}, x \not\models \Box^{\downarrow}({}^0(A))$. Then there is $y$ an $R$-predecessor of $x$, such that $\mathcal{M}, y \not\models {}^0(A)$. And by using the semantic definition of ${}^0(\cdot)$, there is an element $z$, $R$-successor of $y$ such that $\mathcal{M}, z \models A$. But by assumption $\mathcal{M}, x \models (\Diamond A)^0$, that is, in every $R$-predecessor of $x$, $\Diamond A$ is not the case, in particular $\mathcal{M}, y \not\models \Diamond A$, which contradict the fact that $Ryz$ and $\mathcal{M}, z \models A$.

Actually, under the assumption $R_1 = R_2$, including also the duals of $\Diamond$ and $\Box^{\downarrow}$ in the language gives rise to a pleasant symmetry. By defining

$$\begin{array}{rcl} \mathcal{M}, y \models \Diamond^{\downarrow} A & \text{iff} & \exists x.(R_1 xy \;\&\; \mathcal{M}, x \models A) \\ \mathcal{M}, x \models \Box A & \text{iff} & \forall y.(R_1 xy \Rightarrow \mathcal{M}, y \models A), \end{array}$$

we can also prove that ${}^0(\Diamond^{\downarrow} A) \longrightarrow \Box(A^0)$. We can think of $\Diamond$ and $\Diamond^{\downarrow}$ as future ($\mathsf{F}$) and past ($\mathsf{P}$) operators with their duals $\mathsf{H}$ and $\mathsf{G}$ (see [20]), while ${}^0(\cdot)$ and $(\cdot)^0$ behave as a pair of split negations $\sim$ and $\neg$ (see [13, 21]). Among others, all the following properties can be verified.

$$\begin{array}{ll} {\sim}(\mathsf{P}A) \longleftrightarrow \mathsf{G}(\neg A) & \neg(\mathsf{F}A) \longleftrightarrow \mathsf{H}({\sim} A) \\ \mathsf{H}(\neg A) \longleftrightarrow \neg(\mathsf{P}A) & \mathsf{G}({\sim} A) \longleftrightarrow {\sim}(\mathsf{F}A). \end{array}$$

These connections seem to "make sense." In particular, in the last line above, the split negations help us create a connection between the dual modalities $(\mathsf{P}, \mathsf{H})$ and $(\mathsf{F}, \mathsf{G})$. But setting $R_1 = R_2$ seems too strong because we can also prove that, for example,

$$\mathsf{F}A \longrightarrow {\sim}\mathsf{H}{\sim}A$$

which seems to be quite at odds with a "temporal" interpretation of the residuated pairs.

To finish this section we comment on one further possibility. In line both with our previous comment on obtaining $\Diamond$ and $\Box^{\downarrow}$ by "unarizing" the residuated triple, and with the intuitionistic definition of negation in terms of implication and falsum ($\bot$), an interesting alternative can be to connect the unary residuated and Galois connected pairs by means of a ternary relation. We would proceed as follows

**Definition 4.2** Let $\mathcal{M} = \langle W, R \rangle$ be a Kripke model, where $R$ is now a ternary relation on $W$, and let $x, y, z$ be elements in $W$. We define $\models$ as follows

$$
\begin{array}{lll}
x \models \diamond A & \text{iff} & \exists yz.(Rxyz \;\&\; y \models A) \\
y \models \Box^{\downarrow} A & \text{iff} & \forall xz.(Rxyz \Rightarrow x \models A) \\
x \models \Box A & \text{iff} & \forall yz.(Rxyz \Rightarrow y \models A) \\
y \models \diamond^{\downarrow} A & \text{iff} & \exists xz.(Rxyz \;\&\; x \models A) \\
y \models {}^{0}(A) & \text{iff} & \forall xz.(Rxyz \Rightarrow z \not\models A) \\
z \models (A)^{0} & \text{iff} & \forall xy.(Rxyz \Rightarrow y \not\models A).
\end{array}
$$

Intuitively, we have used the first two arguments of the ternary relation to define our residuated operators, while the second and third arguments are used by the Galois connected ones. This gives us a system where the operators interact, but not so strongly as in the previous case. In particular, the following axiom characterizes the interaction present in the system (we take $\bot$ as the always false logic constant, and $\top$ as its dual, true at all points in the model)

$$\mathsf{H}\bot \longleftrightarrow {\sim}\top.$$

Which simply says that $R$ is indeed a *ternary* relation. Having only $\mathsf{H}\bot \longleftrightarrow {\sim}\top$ is probably too weak, but investigating further the connection among the *binary* operators and how this behavior can be incorporated into their unary versions, might lead to further conditions on $R$. This topic deserves deeper reflection, which is beyond the scope of this paper.

## 5 New Derivability Relations

In this section, we discuss possible applications of some of the characteristics of the systems we have been investigating above. The examples we discuss are meant to be taken as tokens of the new features the systems offer.

When working with a logic system as a parser for reasoning with linguistic resources, one of the most important features are the derivability relations among types the proof system can establish. A well known application of this aspect of Lambek calculi is the logic treatment they offer of the *lifting* of *np*, $(np \longrightarrow (s/np)\backslash s)$ first used by Montague [14], and the *value raising principle* (e.g. $np/n \longrightarrow (s/(np\backslash s))/n)$) introduced as a primitive postulate in [19]. If we compare the Definition 3.1 of Galois connections with the inferences used in the analysis of these properties we clearly see that they hinge on the fact that $(/, \backslash)$ is a Galois connected pair. The lifting and value rising properties are indeed instantiations of the composition law for Galois connections.

As pointed out in [17] the composition of Galois connections defines an upwards monotonic function $^*$ which is a *closure* operator satisfying $A \longrightarrow$

$A^*$ and $(A^*)^* \longrightarrow A^*$. The same holds for the composition of residuated pairs. I.e.

$$A \longrightarrow ({}^0(A))^0 \longleftarrow ({}^0(({}^0(A))^0))^0 \qquad A \longrightarrow \Box^\downarrow\Diamond A \longleftarrow \Box^\downarrow\Diamond\Box^\downarrow\Diamond A.$$

Similar derivability relations are used in [4] to account for scope ambiguity phenomena. In particular, the attention is focused on the different scope possibilities of generalized quantifiers (GQs) with respect to negation. This variety among expressions of the same linguistic category is accounted for using the type hierarchy obtained by means of the residuated unary operators. The derivability relation $\Diamond\Box^\downarrow s \longrightarrow s \longrightarrow \Box^\downarrow\Diamond s$ is used to distinguish three different sentential levels: the one lower than negation ($\Diamond\Box^\downarrow s$), the negative one ($s$), and the one higher than negation ($\Box^\downarrow\Diamond s$). The different scope possibilities of generalized quantifiers like *any n*, *a n* and *some n*, are anchored to their type assignments. However, the one dimensional derivability relation given by a pair of residuated operators, may not be enough to account for more intriguing linguistic phenomena, as we will exemplify below in the modeling of GQs sensitive to the polarity of their context.

The proposal presented in [4], does not account for the fact that negative polarity expressions like *any* cannot occur in a positive sentence. We are going to show how Galois operators can be used to solve this problem.

We will consider a linguistic string to be a grammatical sentence if it is proved to be of type $\Box^\downarrow\Diamond s$. We use the following abbreviations: $s_1 := \Diamond\Box^\downarrow s$, $s_2 := s$ and $s_3 := \Box^\downarrow\Diamond s$, viz. $s_1 \longrightarrow s_2 \longrightarrow s_3$, to better visualize the different "sentential levels" encoded in the types. Consider the following type assignments,

$$\begin{array}{ll}
\textit{didn't} & \in (np\backslash s_2)/(np\backslash s_2) \\
\textit{any n} & \in (s_1/np)\backslash s_1 \\
\textit{a n} & \in (s_2/np)\backslash s_2 \\
\textit{some n} & \in (s_3/np)\backslash s_3 \\
\textit{directed} & \in (np\backslash s_1)/np.
\end{array}$$

¿From these type assignments it follows that when parsing a "negative sentence" with a GQ in object position, e.g. *Coppola didn't direct any movie* or *Coppola didn't direct some movie*, the proofs [1a] and [1b] below are obtained, providing the two readings with negation having wide and narrow scope, respectively.

**[1a]** $\neg GQ$

$$\dfrac{\begin{array}{c} np \vdash np \quad np \vdash np \quad s_1 \vdash s_x \quad \boxed{s_y \vdash s_2} \\ \vdots \\ ((np\backslash s_1)/np, (s_x/np)\backslash s_y) \vdash np\backslash s_2 \end{array} \qquad \dfrac{np \vdash np \quad s_2 \vdash s_3}{np, np\backslash s_2 \vdash s_3}[\backslash L]}{\underbrace{np}_{\text{sub}}, (\underbrace{(np\backslash s_2)/(np\backslash s_2)}_{\text{didn't}}, (\underbrace{(np\backslash s_1)/np}_{\text{tv}}, \underbrace{(s_x/np)\backslash s_y}_{\text{GQ}})) \vdash s_3}[/L]$$

**[1b]** $GQ\neg$

$$\dfrac{\begin{array}{c} np \vdash np \quad np \vdash np \quad s_1 \vdash s_2 \quad np \vdash np \quad \boxed{s_2 \vdash s_x} \\ \vdots \\ np, ((np\backslash s_2)/(np\backslash s_2), (np\backslash s_1)/np) \vdash s_x/np \end{array} \qquad s_y \vdash s_3}{\underbrace{np}_{\text{sub}}, (\underbrace{(np\backslash s_2)/(np\backslash s_2)}_{\text{didn't}}, (\underbrace{(np\backslash s_1)/np}_{\text{tv}}, \underbrace{(s_x/np)\backslash s_y}_{\text{GQ}})) \vdash s_3}[\backslash L]$$

When instantiating the GQ with *any movie* the reading [1a] with the GQ in the scope of the negation will be derivable, while the other will fail since $s_x$ will be $s_1$ and $s_2 \not\vdash s_1$. The opposite holds when considering *some movie*: since $s_y = s_3$ the proof in [1a] fails in $s_3 \not\vdash s_2$, while [1b] is derivable.

The idea of stratifying sentential levels by means of derivability relations among types seems promising. However, a linear order between sentential levels doesn't seems to be enough. From the types given above, in fact, it follows that *any movie* can occur in a positive context, e.g. *Coppola directed any movie*, as shown below.

**[2a]**

$$\dfrac{\begin{array}{c} np \vdash np \quad np \vdash np \quad s_1 \vdash s_1 \\ \vdots \\ np, (np\backslash s_1)/np \vdash s_1/np \end{array} \qquad \boxed{s_1 \vdash s_3}}{\underbrace{np}_{\text{Coppola}}, (\underbrace{(np\backslash s_1)/np}_{\text{directed}}, \underbrace{(s_1/np)\backslash s_1}_{\text{any movie}}) \vdash s_3}[\backslash L]$$

Galois connections could offer a solution to this problem by enlarging the type hierarchy as shown below:

$$
\begin{array}{ccccc}
{}^0((\Diamond\Box^{\downarrow}A)^0)) & \longrightarrow & {}^0((A)^0) & \longrightarrow & {}^0((\Box^{\downarrow}\Diamond A)^0) \\
\uparrow & & \uparrow & & \uparrow \\
\Diamond\Box^{\downarrow}s & \longrightarrow & s & \longrightarrow & \Box^{\downarrow}\Diamond s \\
\downarrow & & \downarrow & & \downarrow \\
({}^0(\Diamond\Box^{\downarrow}A))^0 & \longrightarrow & ({}^0(A))^0 & \longrightarrow & ({}^0(\Box^{\downarrow}\Diamond A))^0
\end{array}
$$

We will show, by means of an example, how we could use the type $({}^0(A))^0$ to block the occurrence of negative polarity items in positive sentences. We add a further sentential level $s_4$ encoded by $({}^0(A))^0$ and set

$$
\begin{array}{ll}
\textit{didn't} & \in (np\backslash s_4)/(np\backslash s_2) \\
\textit{any } n & \in (s_1/np)\backslash s_4.
\end{array}
$$

When parsing a sentence like *Coppola directed any movie* starting from these new types the proof in [2a] fails, since $s_4 \not\vdash s_3$. For the same reason, the distribution of *any movie* with respect to negation will be correctly predicted, as the reader can verify by replacing the new lexicon entries in the proofs [1a] and [1b] above.

The type assignments we have been considering so far will not model all the phenomena associated with GQs. Besides accounting for the variety of scope possibilities shown by GQs, a successful modeling should also address the fact that GQs behave syntactically as noun phrases while being able to have semantic scope on a higher sentential level. As shown in [15] this problem cannot be handled without using some kind of structural reasoning. By combining the ideas presented in [15] with the typing system discussed above a complete account is obtained.

This brief (and somewhat naive) linguistic application of Galois operators is intended as a simple example of how the expressiveness of the pure calculus of residuation is extended by adding Galois connections. Notice that this account takes into consideration only the combination of two upwards monotonic functions (the closure operators $\Box^{\downarrow}\Diamond$ and $({}^0(\cdot))^0$). But it could be interesting to explore other possible combinations which a language like $\mathsf{NL}(\Diamond, {}^0(\cdot))$ having both *downwards and upwards monotonic operators* can express.

## 6 Conclusion

As we said in the introduction, the main aim of this paper is to provide new insight on categorial type logics by the use of display calculi. The logic systems encoding residuation we discussed ($\mathsf{NL}$ and $\mathsf{NL}(\diamond)$) are well known in the field, and their meta-logic and proof-theoretic properties have been established long ago. Still, we feel that $\mathsf{DC}$ do provide further insight on how these systems *came to be*, and in which sense they indeed encode in a "pure" state important algebraic properties like residuation and Galois connections. In our analysis, we directly used systems introduced by Goré in [10], our main contribution in the first part of the paper is puzzling out how these systems relate to those standard in the categorial grammar community. Building on our work in Section 2.2, we move on to define the system $\mathsf{NL}(\diamond, {}^0(\cdot))$, having both unary residuated and Galois connected operators. In other words we define a system with operators that resemble a pair of split negations as primitive. In Section 4 we discuss some of the possibilities to build communication among the different operators by a brief detour through semantics. Finally, in Section 5, we provide some ideas on how the systems can be used in analyzing linguistic phenomena.

**Acknowledgment.** In this paper we provide our answers to some questions which were posed to us by other researchers in the field, and we would like to thank them for their inspiration, help and guidance. The work we have done provides the first steps in establishing a connection between display calculi and categorial type logics, something about which Johan van Benthem was wondering about. Our efforts on investigating how to integrate Galois connected operators in $\mathsf{MCTL}$, were kindled by Michael Moortgat. Prof. Moortgat has patiently listened through our first attempts to untangle the issue, and provided important insight. Hopefully, much more about the topic is still to come. We also thank Rajeev Goré for his interest in our interest in display calculi.

## References

[1] K. Ajdukiewicz. Die syntaktische konnexität. *Studia Philosophica*, 1:1–27, 1935.

[2] Y. Bar-Hillel. A quasi-arithmetical notation for syntactic description. *Language*, 29:47–58, 1953.

[3] N. Belnap. Display logic. *Journal of Philosophical Logic*, 11(4):375–417, 1982.

[4] R. Bernardi and R. Moot. Scope ambiguities from a proof-theoretical perspective. In *Proceedings of ICoS-2*, 2000.

[5] G. Birkhoff. *Lattice Theory.* American Mathematical Society, Providence, (1940, 1948, 1967).

[6] T. Blyth and F. Janowitz. *Residuation Theory.* Pergamon Press Inc., New York, 1872.

[7] J. Dunn. Gaggle theory: an abstraction of Galois connections and residuation with applications to negation and various logical operations. In *JELIA 1990: Proceedings of the European Workshop on Logics in Artificial Intelligence*, volume LNCS 478. Springer, 1991.

[8] L. Fuchs. *Partially-ordered algebraic systems.* Pergamon Press Inc., New York, 1963.

[9] R. Goré. Gaggles, Gentzen and Galois: How to display your favourite substructural logic. *Logic Journal of the IGPL*, 6(5):669–694, 1998.

[10] R. Goré. Substructural logics on display. *Logic Journal of the IGPL*, 6(3):451–504, 1998.

[11] J. Lambek. The mathematics of sentence structure. *American Mathematical Monthly*, 65:154–170, 1958.

[12] J. Lambek. On the calculus of syntactic types. In R. Jakobson, editor, *Structure of Languages and Its Mathematical Aspects*, pages 166–178. American Mathematical Society, 1961.

[13] J. Lambek. From categorial to bilinear logic. In K. Došen P. Schröder-Heister, editor, *Substructural Logics.* Oxford University Press, 1993.

[14] R. Montague. *Formal Philosophy: Selected Papers of Richard Montague.* Yale University Press, New Haven, 1974.

[15] M. Moortgat. In situ binding: A modal analysis. In P. Dekker and M. Stokhof, editors, *Proceedings of the Tenth Amsterdam Colloqium*, pages 539–549. ILLC, 1995.

[16] M. Moortgat. Multimodal linguistic inference. *Journal of Logic, Language and Information*, 5(3,4):349–385, 1996.

[17] M. Moortgat. Categorial Type Logics. In J. van Benthem and A. ter Meulen, editors, *Handbook of Logic and Language*, pages 93–178. The MIT Press, Cambridge, Massachusetts, Cambridge, 1997.

[18] R. Moot. Grail: An automated proof assistant for categorial grammar logics. In R. Backhouse, editor, *Proceedings of the 1998 User Interfaces for Theorem Provers Conference*, pages 120–129, 1998.

[19] B. Partee and M. Rooth. Generalized conjunction and type ambiguity. In Bäuerle et al., editor, *Meaning, Use, and Interpretation of Language*, pages 361–383. De Gruyter, Berlin, 1983.

[20] A. Prior. *Past, Present and Future.* Oxford University Press, 1967.

[21] G. Restall. *An introduction to substructural logics.* Routledge, 2000.