Noname manuscript No.
(will be inserted by the editor)

Parsing/Theorem-Proving for Logical Grammar

CatLog3

Glyn Morrill

Received: date / Accepted: date

Abstract *CatLog3* is a 7000 line Prolog parser/theorem-prover for logical categorial grammar. In such logical categorial grammar syntax is universal and grammar is *reduced* to logic: an expression is grammatical if and only if an associated logical statement is a theorem of a fixed calculus. Since the syntactic component is invariant, being the logic of the calculus, logical categorial grammar is purely lexicalist and a particular language model is defined by just a lexical dictionary. The foundational logic of continuity was established by Lambek in 1958 (the Lambek calculus) while a corresponding extension including also logic of discontinuity was established by Morrill and Valentín in 2010 (the displacement calculus). *CatLog3* implements a logic including as primitive connectives the continuous (concatenation) and discontinuous (intercalation) connectives of the displacement calculus, additives, 1st order quantifiers, normal modalities, bracket modalities, and universal and existential subexponentials. In this paper we review the rules of inference for these primitive connectives and their linguistic applications, and we survey the principles of Andreoli's focusing, and of a generalisation of van Benthem's count-invariance, on the basis of which *CatLog3* is implemented.

Keywords count-invariance; focusing; grammar-as-logic; logical categorial grammar; parsing-as-deduction.

The present article is a revised version of Morrill (2017) 'Parsing Logical Grammar: CatLog3', in Loukanova and Liefke (eds.) Proceedings of the Workshop on Logic and Algorithms in Computational Linguistics 2017, LACompLing2017, DiVA, Stockholm University. The research was partially supported by MINECO project TIN2017-89244-R. I thank three anonymous JLLI reviewers for their thoughtful comments, and Oriol Valentín for many related discussions.

G. Morrill
Department of Computer Science
Universitat Politècnica de Catalunya
Jordi Girona Salgado 1–3
Barcelona 08034
Tel.: +34 93 413 7843
Fax: +34 413 7833
E-mail: morrill@cs.upc.edu

1 Introduction

The linguistics that has descended from formal grammar as popularised by Chomsky (1957) has reneged on formalization, and discrete computational grammar in the genre of the 1980s has given way in very large part to statistical NLP. Throughout such changes, however, logical categorial grammar has consistently aspired to practice grammar according to the standards of mathematical logic. The seminal paper in this line is Lambek (1958) which defines the Lambek syntactic calculus and proves Cut-elimination for it, but the tradition dates back further, at least to Bar-Hillel (1953) and Ajdukiewicz (1935). There are a number of monographs and reference articles which expound this logical categorial grammar approach, for example: Moortgat (1988, 1997), Morrill (1994, 2011b), Carpenter (1997), Jäger (2005), and Moot and Retoré (2012).

The Lambek calculus **L** is a calculus of concatentation which is free of structural rules, and which enjoys Cut-elimination and its corollaries — the subformula property, decidability, the finite reading property, and the focusing property:

- The **Cut-elimination property** for a calculus is the property whereby every theorem of the calculus has a Cut-free proof; for the case of **L** see the appendix of Lambek (1958);
- The **subformula property** is the property whereby every theorem has a proof containing only its subformulas; this usually follows, as it does in the case of **L**, from Cut-elimination plus the fact that in all other rules, all the formulas in the premises are proper or improper subformulas of formulas in the conclusions;
- The **decidability property** is the property whereby the characteristic function of the set of sequents which are theorems is computable; this follows in the case of **L** from the facts of Cut-elimination plus the finiteness of the backward-chaining Cut-free sequent proof search space;
- The **finite reading property** is the property whereby no expression in the language model defined by a grammar has an infinite number of readings; this follows in the case of **L** essentially from the fact that Cut-elimination is *semantic* — Cut is semantically interpreted by substitution in such a way that elimination of Cut from a proof conserves the compositional reading (Hendriks, 1993); the finite reading property follows because: 1/. the lexicon is finite so lexical readings cannot be a source of infinite ambiguity; 2/. the search-space of backward-chaining Cut-free sequent proof search is finite, and so there are only a finite number of Cut-free derivational readings; and 3/. by the semanticity of Cut-elimination a theorem's proofs using Cut, although infinite in number, add no derivational readings over and above those of the Cut-free proofs, which are finite in number.
- The **focusing property** is a property of Cut-free backward-chaining sequent proof search discovered by J.M. Andreoli: the founding fathers of 20th century logic knew about invertible rules but Andreoli showed, in the context of linear logic, that dual to invertible rules there are focusing rules which lock onto noninvertible formulas (focusing was unanticipated even by the founder of linear logic J.Y. Girard).

The displacement calculus of Morrill and Valentín (2010) and Morrill et al. (2011) extends Lambek calculus with intercalation. The displacement calculus **D** contains both continuous and discontinuous connective families, while remaining free of structural rules, and enjoying Cut-elimination and its good corollaries, in the same way as **L**: the subformula property, decidability, the finite reading property, and the focusing property.

Although it features discontinuity, which appears to be partially non-commutative in nature, displacement calculus **D** in fact has non-commutative formulations of both continuity

and discontinuity parallel to that of Lambek calculus **L** due to an innovative 'h-sequent' calculus ('h' for 'hyper' or 'hedge'). This greatly facilitates the proof of Cut-elimination for the displacement calculus (Morrill et al., 2011; Valentín, 2012). It also enables logic programming of displacement calculus theorem-proving; relevant data structures and techniques are given in Morrill (2011a). On the basis of this the *CatLog* program series comprises implementations in Prolog of logical categorial grammar parser/theorem-provers including:

- *CatLog1* (Morrill, 2012), based on the method of uniform proof (Miller et al., 1991), and on count-invariance for multiplicatives (van Benthem, 1991);
- *CatLog2*, based on Andreoli's method of focusing (Andreoli, 1992), and on count-invariance for multiplicatives, additives and bracket modalities (Valentín et al., 2013);
- *CatLog3*, based on the focalization of Andreoli, and count-invariance for multiplicatives, additives, and exponentials (Kuznetsov et al., 2017).

In this paper we survey the principles on which *CatLog3* is based. The structure is as follows. In Section 2 we present the primitive connectives of the logical fragment for which parsing/theorem-proving is implemented and describe some of their linguistic applications. In Section 3 we discuss focusing. In Section 4 we discuss count-invariance. In Section 5 we evaluate *CatLog3* compared to *CatLog2*. In Section 6 we evaluate further comparison in relation to, for example, the *Montague Test* (Morrill and Valentín, 2016): the task of providing a computational cover grammar of Montague's Montague (1973) grammar fragment. We conclude in Section 7. The appendix comprises unedited *CatLog3* LaTeX output for relativisation examples.

2 Displacement logic

The formalism used by *CatLog3* contains the connectives of Table 1. The heart of the logic is the displacement calculus made up of twin continuous and discontinuous residuated families of connectives having a pure Gentzen sequent calculus — without prosodic labelling, free of structural rules, and enjoying Cut-elimination. Other primary connectives include additives, 1st order quantifiers, normal (i.e. distributive) modalities, bracket (i.e. nondistributive) modalities, and subexponentials.

Let $A \Rightarrow B$ be a logical statement asserting that in every interpretation the meaning of A is a subset of the meaning of B. In any logical system on such a design the reflexivity $A \Rightarrow A$ is valid by the reflexivity of set containment and the transitivity $A \Rightarrow B \ \& \ B \Rightarrow C \ / \ A \Rightarrow C$ is valid by the transitivity of set containment. In calculi the identity axiom embodies reflexivity and the Cut rule transitivity. But while the identity axiom is harmless and necessary, the Cut rule is problematic in that it introduces a new unknown type (the Cut formula) reading from conclusion to premises. In order to have the effect of Cut without actually having to use it we can show that Cut is *eliminable* (conserving theorems) from a Cut-based system or show that Cut is *admissible* (conserving theorems) to a Cut-free system.

Constructive, combinatoric, proofs of Cut-elimination have deep proof-theoretical and computational meanings. But a combinatoric proof of Cut-elimination for all the connectives here would need to treat very many cases. Furthermore, for present purposes all that is required to justify Cut-free proof search is that Cut be admissible to a Cut-free formulation. Therefore, in work on progress O. Valentín is investigating algebraic semantics for the con-

	cont. mult.	disc. mult.	add.	qu.	norm. mod.	brack. mod.	subexp.	limited contr. & weak.
primary	$/$ $\backslash$ $\bullet$ I	$\uparrow$ $\downarrow$ $\odot$ J	$\&$ $\oplus$	$\bigwedge$ $\bigvee$	$\Box$ $\Diamond$	$[\,]^{-1}$ $\langle\rangle$	! ?	| W
sem. inactive variants	$\bullet\!\!-\!\!\circ$ $\circ\!\!-\!\!\bullet$ ◐ ◑	↾ ⇂ ↿ ⇃ ◓ ◒	$\sqcap$ $\sqcup$	$\forall$ $\exists$	■ ♦			
det.	$\triangleleft^{-1}$ $\triangleright^{-1}$	ˇ						
synth.	$\triangleleft$ $\triangleright$	ˆ						
nondet.	$\div$	$\Uparrow$ $\Downarrow$						
synth.	$\times$	◎						

Table 1 Categorial connectives

nectives of *CatLog3*, for which completness results have as a corollary (non-constructive) 'semantic' Cut-elimination = Cut-admissibility.[1]

We can draw a clear distinction between the primary connectives, the semantically inactive connectives, and the synthetic connectives; the latter two are there only for convenience, to abbreviate types and to simplify derivations. There are semantically inactive variants of the continuous and discontinuous multiplicatives, and semantically inactive variants of the additives, 1st order quantifiers, and normal modalities.[2] Synthetic connectives (Girard, 2011) divide into the continuous and discontinuous deterministic (unary) synthetic and nondeterministic (binary) synthetic connectives.[3]

2.1 Syntactic types

The syntactic types of displacement logic are prosodically sorted $\mathcal{F}_0, \mathcal{F}_1, \mathcal{F}_2, \ldots$ according to the number of points of discontinuity $0, 1, 2, \ldots$ that their expressions contain. Each syntactic type predicate letter has an arity and a prosodic sort which are naturals, and a corresponding

[1] Once Cut-elimination/admissibility has been established, the only challenge to decidability here comes from non-linearity: the infinitary left rule of the existential subexponential (Buszkowski, 2007; Buszkowski and Palka, 2008); and the contraction rule of the universal subexponential. In this connection, linguistically the existential subexponential left rule is not required. But the contraction rule of the universal subexponential *is* required, for parasitic gaps. The Lambek calculus $\mathbf{Lb!_b}$ with bracket modalities and the universal subexponential bracket conditioned contraction rule of Morrill (2017) is undecidable (Kanovich et al., 2017). However *CatLog3* uses a different bracket-conditioned contraction rule for the universal subexponential, essentially that of Morrill (2011b).

[2] For example, the semantically inactive additive conjunction $A \sqcap B\colon \phi$ abbreviates $A \& B\colon (\phi, \phi)$.

[3] For example, the nondeterministic continuous division $B \div A\colon \phi$ abbreviates $(A \backslash B) \sqcap (B / A)\colon \phi$, which is to say $(A \backslash B) \& (B / A)\colon (\phi, \phi)$.

1.	$\mathcal{F}_i ::= \mathcal{F}_{i+j}/\mathcal{F}_j$	$T(C/B)$	$= T(B)\rightarrow T(C)$	over
2.	$\mathcal{F}_j ::= \mathcal{F}_i\backslash\mathcal{F}_{i+j}$	$T(A\backslash C)$	$= T(A)\rightarrow T(C)$	under
3.	$\mathcal{F}_{i+j} ::= \mathcal{F}_i\bullet\mathcal{F}_j$	$T(A\bullet B)$	$= T(A)\&T(B)$	continuous product
4.	$\mathcal{F}_0 ::= I$	$T(I)$	$= \top$	continuous unit
5.	$\mathcal{F}_{i+1} ::= \mathcal{F}_{i+j}\uparrow_k\mathcal{F}_j, 1 \leq k \leq i+j$	$T(C\uparrow_k B)$	$= T(B)\rightarrow T(C)$	circumfix
6.	$\mathcal{F}_j ::= \mathcal{F}_{i+1}\downarrow_k\mathcal{F}_{i+j}, 1 \leq k \leq i+1$	$T(A\downarrow_k C)$	$= T(A)\rightarrow T(C)$	infix
7.	$\mathcal{F}_{i+j} ::= \mathcal{F}_{i+1}\odot_k\mathcal{F}_j, 1 \leq k \leq i+1$	$T(A\odot_k B)$	$= T(A)\&T(B)$	discontinuous product
8.	$\mathcal{F}_1 ::= J$	$T(J)$	$= \top$	discontinuous unit
9.	$\mathcal{F}_i ::= \mathcal{F}_i\&\mathcal{F}_i$	$T(A\&B)$	$= T(A)\&T(B)$	additive conjunction
10.	$\mathcal{F}_i ::= \mathcal{F}_i\oplus\mathcal{F}_i$	$T(A\oplus B)$	$= T(A)+T(B)$	additive disjunction
11.	$\mathcal{F}_i ::= \bigwedge V\mathcal{F}_i$	$T(\bigwedge vA)$	$= F\rightarrow T(A)$	1st order univ. qu.
12.	$\mathcal{F}_i ::= \bigvee V\mathcal{F}_i$	$T(\bigvee vA)$	$= F\&T(A)$	1st order exist. qu.
13.	$\mathcal{F}_i ::= \Box\mathcal{F}_i$	$T(\Box A)$	$= \mathbf{L}T(A)$	universal modality
14.	$\mathcal{F}_i ::= \Diamond\mathcal{F}_i$	$T(\Diamond A)$	$= \mathbf{M}T(A)$	existential modality
15.	$\mathcal{F}_i ::= [\,]^{-1}\mathcal{F}_i$	$T([\,]^{-1}A)$	$= T(A)$	univ. bracket modality
16.	$\mathcal{F}_i ::= \langle\rangle\mathcal{F}_i$	$T(\langle\rangle A)$	$= T(A)$	exist. bracket modality
17.	$\mathcal{F}_0 ::= !\mathcal{F}_0$	$T(!A)$	$= T(A)$	universal subexponential
18.	$\mathcal{F}_0 ::= ?\mathcal{F}_0$	$T(?A)$	$= T(A)^+$	existential subexponential

Table 2 Syntactic types of our fragment of displacement logic

semantic type. Assuming ordinary feature terms to be given, where P is a type predicate letter of arity n and prosodic sort i, and $t_1, \ldots, t_n$ are feature terms, $Pt_1 \ldots t_n$ is an atomic type of prosodic sort i and of the semantic type corresponding to P. Compound types of our fragment of displacement logic are formed as illustrated in Table 2; the Backus-Naur grammar in the lefthand column has one clause per connective and is regulated by the prosodic sorts of syntactic types; the structure preserving semantic type map T in the righthand column associates these syntactic types with their semantic types.

2.2 Gentzen sequent calculus

We use for displacement logic a variant 'h-sequent' calculus of the Gentzen sequent presentations familiar from Gentzen (1934) and Lambek (1958). The letter h can be seen as standing for 'hyper', because the notation of the calculus invokes a kind of intercalation of configurations within configurations, like hypertext. It can also be seen as standing for 'hedge', because structurally this intercalation of configurations is an instance of the data structure known as a hedge.

In Gentzen sequent antecedents for displacement logic with bracket modalities (structural inhibition) and exponentials (structural facilitation) there are also bracket constructors and 'stoups'.

Stoups (cf. the linear logic of Girard (2011) (ζ) are stores which are read as multisets for re-usable (non-linear) resources. A *zone* (Ξ) comprises a configuration together with a stoup. The stoup appears at the left of the configuration marked off by a semicolon (when the stoup is empty the semicolon may be omitted). The stoup of linear logic is for resources which can be contracted (copied) or weakened (deleted). Whereas linear logic is commutative, our logic is non-commutative and here the stoup is used for resources which are commutative. Furthermore, our stoup is for a linguistically motivated bracket-conditioned variant of con-

traction, and does not allow weakening.[4] The bracket constructor applies to a configuration together with a stoup, i.e. to a zone, and reusable resources are specific to their domain. Stoups $\mathcal{S}$, configurations $\mathcal{O}$ and tree terms $\mathcal{T}$ are defined by the following, where $\emptyset$ is the metalinguistic empty stoup, Λ is the metalinguistic empty configuration, and the *separator* 1 marks points of discontinuity:

(1) a. $\mathcal{S} ::= \emptyset \mid \mathcal{F}_0, \mathcal{S}$
b. $\mathcal{O} ::= \Lambda \mid \mathcal{T}, \mathcal{O}$
c. $\mathcal{T} ::= 1 \mid \mathcal{F}_0 \mid \mathcal{F}_{i>0}\{\underbrace{\mathcal{O} : \ldots : \mathcal{O}}_{i\,\mathcal{O}'\text{s}}\} \mid [\mathcal{S}; \mathcal{O}]$

For a type A, its sort $s(A)$ is the i such that $A \in \mathcal{F}_i$. For a configuration Γ, its sort $s(\Gamma)$ is $|\Gamma|_1$, i.e. the number of points of discontinuity 1 which it contains. We see in (1c) the key feature of the configurations of h-sequent calculus for displacement which is that while types continuous types (sort zero) are terminal, discontinuous types (sort non-zero) are non-terminal and dominate intercalated configurations (recursively).

Sequents are of the form:

(2) $\mathcal{S}; \mathcal{O} \Rightarrow \mathcal{F}$ such that $s(\mathcal{O}) = s(\mathcal{F})$

Note from (1a) that only types of sort 0 go into the stoup; reusable types of other sorts would not preserve the sequent antecedent-succedent sort equality of (2) under contraction or expansion since $i + i = i$ for $i = 0$, but $i + i \neq i$ for $i \neq 0$.

The figure $\overrightarrow{A}$ of a type A is defined by:

(3) $\overrightarrow{A} = \begin{cases} A & \text{if } s(A) = 0 \\ A\{\underbrace{1 : \ldots : 1}_{s(A)\ 1'\text{s}}\} & \text{if } s(A) > 0 \end{cases}$

The figure of a type is a kind of 'eta-long' form which structurally reflects the sort of the type. In h-sequent calculus rules, the antecedent types to which inference applies (the *active* types) are always such figures, meshing the sorted types structurally with their sorted structural context.

Where Ξ and Ξ' are configurations or zones, the distinguished occurrence notation $\Xi(\Xi')$ represents a distinguished occurrence of Ξ' in context Ξ. Where Γ is a configuration of sort i and $\Delta_1, \ldots, \Delta_i$ are configurations, the *fold* $\Gamma \otimes \langle \Delta_1 : \ldots : \Delta_i \rangle$ is the result of replacing the successive 1's in Γ by $\Delta_1, \ldots, \Delta_i$ respectively. Where Γ is of sort i, the hyperoccurrence notation $\Delta\langle\Gamma\rangle$ abbreviates $\Delta_0(\Gamma \otimes \langle \Delta_1 : \ldots : \Delta_i \rangle)$, i.e. a context configuration Δ (which is externally Δ_0 and internally $\Delta_1, \ldots, \Delta_i$) with a potentially discontinuous distinguished subconfiguration Γ (continuous if $i = 0$; discontinuous if $i \neq 0$). Where Γ is a configuration $\langle\Gamma\rangle$ represents Γ_0 such that $\Gamma_0\langle\Delta_1 : \ldots : \Delta_n\rangle = \Gamma$ where $\Delta_1, \ldots, \Delta_n$ are configurations.

Where Δ is a configuration of sort $i > 0$ and Γ is a configuration, the kth *metalinguistic intercalation* $\Delta \mid_k \Gamma$, $1 \leq k \leq i$, is given by:

(4) $\Delta \mid_k \Gamma =_{df} \Delta \otimes \langle \underbrace{1 : \ldots : 1}_{k-1\ 1\text{'s}} : \Gamma : \underbrace{1 : \ldots : 1}_{i-k\ 1\text{'s}} \rangle$

i.e. $\Delta \mid_k \Gamma$ is the configuration that results from replacing by Γ the kth separator in Δ. Just as ',' in (1b) is the continuous metalinguistic structural constructor (concatenation/appending), '$\mid_k$' is the discontinuous metalinguistic structural constructor (intercalation/plugging).

[4] To anticipate linguistic analysis a little, a hypothetical gap subtype emitted by a relative pronoun corresponding to a long-distance dependency will enter a stoup, percolate in stoups, may contract to create (parasitic) gaps, and finally permutes into a (host) extraction site.

2.3 Rules and linguistic applications

In this section we give rules which are both syntactic and semantic. A semantically labelled sequent is a sequent in which the antecedent type occurrences $A_1, \ldots, A_n$ are labelled by distinct semantic variables $x_1, \ldots, x_n$ which are of types $T(A_1), \ldots, T(A_n)$ respectively, and the succedent type A is labelled by a semantic term $\phi(x_1, \ldots, x_n)$ of type $T(A)$ with free variables drawn from $x_1, \ldots, x_n$. As well as giving the semantically labelled Gentzen h-sequent rules for the primitive connectives of our *CatLog3* fragment of displacement logic, we illustrate their linguistic applications.

In h-sequents the active types in antecedents are figures (vectorial) whereas those in succedents are not; intuitively this is because antecedents are structured but succedents are not. Otherwise, apart from the stoups, the rules of all of the connectives have the same basic shapes as the usual (continuous) rules of categorial logic but including angle brackets distinguishing discontinuous substructures as well as round brackets distinguishing continuous substructures.

1. $$\frac{\zeta_1; \Gamma \Rightarrow B\colon \psi \qquad \Xi(\zeta_2; \Delta_1, \langle \overrightarrow{C}\colon z \rangle, \Delta_2) \Rightarrow D\colon \omega}{\Xi(\zeta_1 \uplus \zeta_2; \Delta_1, \langle \overrightarrow{C/B}\colon x, \Gamma \rangle, \Delta_2) \Rightarrow D\colon \omega\{(x\ \psi)/z\}}\ /L \qquad \frac{\zeta; \Gamma, \overrightarrow{B}\colon y \Rightarrow C\colon \chi}{\zeta; \Gamma \Rightarrow C/B\colon \lambda y \chi}\ /R$$

2. $$\frac{\zeta_1; \Gamma \Rightarrow A\colon \phi \qquad \Xi(\zeta_2; \Delta_1, \langle \overrightarrow{C}\colon z \rangle, \Delta_2) \Rightarrow D\colon \omega}{\Xi(\zeta_1 \uplus \zeta_2; \Delta_1, \langle \Gamma, \overrightarrow{A\backslash C}\colon y \rangle, \Delta_2) \Rightarrow D\colon \omega\{(y\ \phi)/z\}}\ \backslash L \qquad \frac{\zeta; \overrightarrow{A}\colon x, \Gamma \Rightarrow C\colon \chi}{\zeta; \Gamma \Rightarrow A\backslash C\colon \lambda x \chi}\ \backslash R$$

3. $$\frac{\Xi\langle \overrightarrow{A}\colon x, \overrightarrow{B}\colon y \rangle \Rightarrow D\colon \omega}{\Xi\langle \overrightarrow{A \bullet B}\colon z \rangle \Rightarrow D\colon \omega\{\pi_1 z/x, \pi_2 z/y\}}\ \bullet L \qquad \frac{\zeta_1; \Delta \Rightarrow A\colon \phi \qquad \zeta_2; \Gamma \Rightarrow B\colon \psi}{\zeta_1 \uplus \zeta_2; \Delta, \Gamma \Rightarrow A \bullet B\colon (\phi, \psi)}\ \bullet R$$

4. $$\frac{\Xi\langle \Lambda \rangle \Rightarrow A\colon \phi}{\Xi\langle \overrightarrow{I}\colon x \rangle \Rightarrow A\colon \phi}\ IL \qquad \frac{}{\emptyset; \Lambda \Rightarrow I\colon 0}\ IR$$

Fig. 1 Lambek multiplicatives

The continuous multiplicatives, the Lambek connectives of Lambek (1958, 1988) are given in Figure 1. The Lambek connectives, defined in relation to concatenation/appending, are the basic means of categorial categorization and subcategorization. The directional divisions over, /, and under, \, are exemplified by assignments such as *the*: N/CN for *the man*: N, and *sings*: $N\backslash S$ for *John sings*: S, *loves*: $(N\backslash S)/N$ for *John loves Mary*: S.

The discontinuous multiplicatives of Figure 2, the displacement connectives (Morrill and Valentín, 2010; Morrill et al., 2011) are defined in relation to intercalation/plugging. In the discontinuous rules the value k of the subindex indicates that the intercalation of the inference occurs at the kth point of discontinuity counting from the left; it defaults to the first position, i.e. under omission it is to be taken to be 1. Circumfixation, $\uparrow$, is exemplified by a discontinuous particle verb assignment such as *calls*+1+*up*: $(N\backslash S)\uparrow N$ for *Mary calls John up*: S, and infixation, $\downarrow$, and circumfixation together are exemplified by a quantifier phrase assignment of the form *everyone*: $(S\uparrow N)\downarrow S$ simulating Montague's S14 treatment of quantifying in (see Section 6).

In relation to the Lambek and displacement rules, notice that the stoup is partitioned between the two premises reading bottom-up from conclusions to premises in the case of

5. $$\frac{\zeta_1;\Gamma \Rightarrow B:\psi \qquad \Xi(\zeta_2;\Delta_1,\langle \overrightarrow{C}:z\rangle,\Delta_2) \Rightarrow D:\omega}{\Xi(\zeta_1 \uplus \zeta_2;\Delta_1,\langle \overrightarrow{C\uparrow_k B}:x\,|_k\,\Gamma\rangle,\Delta_2) \Rightarrow D:\omega\{(x\,\psi)/z\}}\uparrow_k L \qquad \frac{\zeta;\Gamma\,|_k\,\overrightarrow{B}:y \Rightarrow C:\chi}{\zeta;\Gamma \Rightarrow C\uparrow_k B:\lambda y\chi}\uparrow_k R$$

6. $$\frac{\zeta_1;\Gamma \Rightarrow A:\phi \qquad \Xi(\zeta_2;\Delta_1,\langle \overrightarrow{C}:z\rangle,\Delta_2) \Rightarrow D:\omega}{\Xi(\zeta_1 \uplus \zeta_2;\Delta_1,\langle \Gamma\,|_k\,\overrightarrow{A\downarrow_k C}:y\rangle,\Delta_2) \Rightarrow D:\omega\{(y\,\phi)/z\}}\downarrow_k L \qquad \frac{\zeta;\overrightarrow{A}:x\,|_k\,\Gamma \Rightarrow C:\chi}{\zeta;\Gamma \Rightarrow A\downarrow_k C:\lambda x\chi}\downarrow_k R$$

7. $$\frac{\Xi\langle \overrightarrow{A}:x\,|_k\,\overrightarrow{B}:y\rangle \Rightarrow D:\omega}{\Xi\langle \overrightarrow{A\odot_k B}:z\rangle \Rightarrow D:\omega\{\pi_1 z/x,\pi_2 z/y\}}\odot_k L \qquad \frac{\zeta_1;\Delta \Rightarrow A:\phi \qquad \zeta_2;\Gamma \Rightarrow B:\psi}{\zeta_1 \uplus \zeta_2;\Delta\,|_k\,\Gamma \Rightarrow A\odot_k B:(\phi,\psi)}\odot_k R$$

8. $$\frac{\Xi\langle 1\rangle \Rightarrow A:\phi}{\Xi\langle \overrightarrow{J}:x\rangle \Rightarrow A:\phi}JL \qquad \frac{}{\emptyset;1 \Rightarrow J:0}JR$$

Fig. 2 Displacement multiplicatives

binary multiplicative rules, copied to the premise in the case of unary multiplicative rules, and empty in the case of nullary multiplicative rules (axioms).

9. $$\frac{\Xi\langle \overrightarrow{A}:x\rangle \Rightarrow C:\chi}{\Xi\langle \overrightarrow{A\&B}:z\rangle \Rightarrow C:\chi\{\pi_1 z/x\}}\&L_1 \qquad \frac{\Xi\langle \overrightarrow{B}:y\rangle \Rightarrow C:\chi}{\Xi\langle \overrightarrow{A\&B}:z\rangle \Rightarrow C:\chi\{\pi_2 z/y\}}\&L_2$$

$$\frac{\Xi \Rightarrow A:\phi \qquad \Xi \Rightarrow B:\psi}{\Xi \Rightarrow A\&B:(\phi,\psi)}\&R$$

10. $$\frac{\Xi\langle \overrightarrow{A}:x\rangle \Rightarrow C:\chi_1 \qquad \Xi\langle \overrightarrow{B}:y\rangle \Rightarrow C:\chi_2}{\Xi\langle \overrightarrow{A\oplus B}:z\rangle \Rightarrow C:z \to x,\chi_1;y,\chi_2}\oplus L$$

$$\frac{\Xi \Rightarrow A:\phi}{\Xi \Rightarrow A\oplus B:\iota_1\phi}\oplus R_1 \qquad \frac{\Xi \Rightarrow B:\psi}{\Xi \Rightarrow A\oplus B:\iota_2\psi}\oplus R_2$$

Fig. 3 Additives

The additives of Figure 3, polymorphic connectives (Lambek, 1961; Kanazawa, 1992; Morrill, 1994) have application to weak polymorphism. For example, the additive conjunction & can be used for the polymorphism *rice*: $N\&CN$ as in *rice grows*: S and *the rice grows*: S,[5] and the additive disjunction $\oplus$ can be used for the polymorphism *is*: $(N\backslash S)/(N\oplus(CN/\,CN)$ as in *Tully is Cicero*: S and *Tully is humanist*: S.

Notice how the stoup is identical in conclusions and premises of additive rules.

The quantifiers of Figure 4 (Morrill, 1994) have application to features. For example, singular and plural number in *sheep*: $\bigwedge nCNn$ for *the sheep grazes*: S and *the sheep graze*: S. And for a past, present or future tense finite sentence complement, *said*: $(N\backslash S)/\bigvee tS\,f(t)$ in *John said Mary walked*: S, *John said Mary walks*: S, and *John said Mary will walk*: S.

Notice how the stoup is identical in conclusion and premise in each quantifier rule.

[5] Note the advantages of such polymorphism over assuming empty operators: if say empty determiners were allowed they could a priori occur any number of times in any positions; and they would also overgenerate, for example, the ungrammatical **most or dogs* on the pattern of the grammatical *most or all dogs*.

$$11.\quad \frac{\Xi\langle\overrightarrow{A[t/v]}\colon x\rangle \Rightarrow B\colon\psi}{\Xi\langle\overrightarrow{\bigwedge vA}\colon z\rangle \Rightarrow B\colon\psi\{(z\ t)/x\}}\bigwedge L \qquad \frac{\Xi \Rightarrow A[a/v]\colon\phi}{\Xi \Rightarrow \bigwedge vA\colon\lambda v\phi}\bigwedge R^{\dagger}$$

$$12.\quad \frac{\Xi\langle\overrightarrow{A[a/v]}\colon x\rangle \Rightarrow B\colon\psi}{\Xi\langle\overrightarrow{\bigvee vA}\colon z\rangle \Rightarrow B\colon\psi\{\pi_2 z/x\}}\bigvee L^{\dagger} \qquad \frac{\Xi \Rightarrow A[t/v]\colon\phi}{\Xi \Rightarrow \bigvee vA\colon(t,\phi)}\bigvee R$$

Fig. 4 Quantifiers, where † indicates that there is no a in the conclusion

$$13.\quad \frac{\Xi\langle\overrightarrow{A}\colon x\rangle \Rightarrow B\colon\psi}{\Xi\langle\overrightarrow{\Box A}\colon z\rangle \Rightarrow B\colon\psi\{^{\vee}z/x\}}\Box L \qquad \frac{\boxtimes\Xi \Rightarrow A\colon\phi}{\boxtimes\Xi \Rightarrow \Box A\colon{}^{\wedge}\phi}\Box R$$

$$14.\quad \frac{\boxtimes\Xi\langle\overrightarrow{A}\colon x\rangle \Rightarrow \diamondsuit B\colon\psi}{\boxtimes\Xi\langle\overrightarrow{\diamondsuit A}\colon z\rangle \Rightarrow \diamondsuit B\colon\psi\{^{\cup}z/x\}}\diamondsuit L \qquad \frac{\Xi \Rightarrow A\colon\phi}{\Xi \Rightarrow \diamondsuit A\colon{}^{\cap}\phi}\diamondsuit R$$

Fig. 5 Normal modalities, where $\boxtimes/\diamondsuit$ marks a structure all the types of which have main connective a box/diamond

With respect to the (**S4**) normal modalities of Figure 5, the universal (Morrill, 1990) has application to intensionality. For example, for a propositional attitude verb such as *believes* we can assign type $\Box((N\backslash S)/\Box S)$ with a modality outermost since the word has an intension, and a modality on the first argument but not the second, since the sentential complement is an intensional domain, but not the subject. Note that the modalities are in the categorial type: distinctly from, but in relation to, the logical semantics of the propositional attitude verb. The $\Box$ Right rule is semantically interpreted by intensionalisation $^{\wedge}$ and the $\Box$ Left rule is semantically interpreted by extensionalisation $^{\vee}$ in such a way that the Curry-Howard proof detour normalization correspondence for the modality yields the law of down-up cancellation (Dowty et al., 1981): $^{\vee\wedge}\phi = \phi$.

Notice how the stoup is identical in conclusion and premise in each normal modality rule.

$$15.\quad \frac{\Xi\langle\overrightarrow{A}\colon x\rangle \Rightarrow B\colon\psi}{\Xi\langle[\overrightarrow{\ ]^{-1}A}\colon x]\rangle \Rightarrow B\colon\psi}[\ ]^{-1}L \qquad \frac{[\Xi] \Rightarrow A\colon\phi}{\Xi \Rightarrow [\ ]^{-1}A\colon\phi}[\ ]^{-1}R$$

$$16.\quad \frac{\Xi\langle[\overrightarrow{A}\colon x]\rangle \Rightarrow B\colon\psi}{\Xi\langle\overrightarrow{\langle\rangle A}\colon x\rangle \Rightarrow B\colon\psi}\langle\rangle L \qquad \frac{\Xi \Rightarrow A\colon\phi}{[\Xi] \Rightarrow \langle\rangle A\colon\phi}\langle\rangle R$$

Fig. 6 Bracket modalities

The bracket modalities of Figure 6, Moortgat (1996) and Morrill (1992), have application to non-associativity and syntactical domains such as extraction islands and prosodic phrases. For example, single bracketing for weak islands may take the form *walks*: $\langle\rangle N\backslash S$ for the subject condition, and *without*: $[\ ]^{-1}(VP\backslash VP)/VP$ for the adverbial island constraint; and there may be double bracketing for strong islands such as *and*: $(S\backslash[\ ]^{-1}[\ ]^{-1}S)/S$ for the coordinate structure constraint.

Notice how the stoup is identical in conclusions and premises of bracket modality rules.

17.
$$\frac{\Xi(\zeta \uplus \{A\colon x\}; \Gamma_1, \Gamma_2) \Rightarrow B\colon \psi}{\Xi(\zeta; \Gamma_1, !A\colon x, \Gamma_2) \Rightarrow B\colon \psi}\, !L \qquad \frac{!A\colon x \Rightarrow B\colon \phi}{!A\colon x \Rightarrow !B\colon \phi}\, !R$$

$$\frac{\Xi(\zeta; \Gamma_1, A\colon x, \Gamma_2) \Rightarrow B\colon \psi}{\Xi(\zeta \uplus \{A\colon x\}; \Gamma_1, \Gamma_2) \Rightarrow B\colon \psi}\, !P$$

$$\frac{\Xi(\{A\colon x\}; \Gamma_1, [\{A\colon y\}; \Gamma_2], \Gamma_3) \Rightarrow B\colon \psi}{\Xi(\zeta \uplus \{A\colon x\}; \Gamma_1, [[\Gamma_2]], \Gamma_3) \Rightarrow B\colon \psi\{x/y\}}\, !C$$

18.
$$\frac{\Xi(A\colon x_1) \Rightarrow B\colon \psi([x_1]) \qquad \Xi(A\colon x_1, A\colon x_2) \Rightarrow B\colon \psi([x_1, x_2]) \qquad \dots}{\Xi(?A\colon x) \Rightarrow B\colon \psi(x)}\, ?L$$

$$\frac{\Xi \Rightarrow A\colon \phi}{\Xi \Rightarrow ?A\colon [\phi]}\, ?R \qquad \frac{\zeta; \Gamma \Rightarrow A\colon \phi \qquad \zeta'; \Delta \Rightarrow ?A\colon \psi}{\zeta \uplus \zeta'; \Gamma, \Delta \Rightarrow ?A\colon [\phi|\psi]}\, ?E$$

Fig. 7 Subexponentials

Finally, there are the non-linear subexponentials ! for contraction and ? for expansion of Figure 7, these originating from the universal and existential exponentials of the linear logic of Girard (1987). The formulations of $!R$ and $!C$ given in Figure 7 and used in *CatLog3* differ from Morrill (2017) and from previous versions: $!R$, which has a single rather than multiple antecedent types, is based on the pattern of 'soft' linear logic (Lafont, 2004);[6] $!C$, which has double bracketing in the contraction domain in the conclusion, is based on (though not quite identical to) the pattern of the contraction rule developed in Morrill (2011b).

The universal subexponential ! has application to extraction, including parasitic extraction. We can assign a relative pronoun type *that*: $(CN\backslash CN)/(S/!N)$ (or *that*: $(CN\backslash CN)/(!N\backslash S)$: since $!A$ permutes, it makes no difference). The rule $!L$ moves the operand of a universal subexponential such as the hypothetical subtype $!N$ of relativization into the stoup, where it will percolate according to all the other rules. An eventual application of $!P$ can move the hypothetical subtype of relativization into a (non-island) extraction site. Crucially in the linguistic formulation of subexponentials, and unlike in Girard's original linear logic exponentials, ! does not have weakening, i.e. deletion, since, for example, the body of a relative clause *must* contain a gap: **man who John loves Mary*.

Using the universal subexponential !, for which contraction, reading from conclusion to premise, converts double island brackets to single island brackets, the relative pronoun type *that*: $(CN\backslash CN)/(S/!N)$ also allows parasitic extraction such as that in *man that the friends of admire* or that in *paper that John filed without reading*, where parasitic gaps can appear only in weak islands, but can iterate indefinitely in subislands as in for example *man who the fact that the friends of admire without praising surprises*.

The expansion existential subexponential ? has application to iterated coordination (Morrill, 1994). Using the existential subexponential we can assign a coordinator type such as *and*: $(?N\backslash N)/N$ allowing the iterated coordination of, for example, *John, Bill, Mary and Suzy*: N, or such as *and*: $(?(S/N)\backslash(S/N))/(S/N)$ for, say, *John dislikes*, *Mary likes, and Bill loves, London* (iterated right node raising), and so on.

In relation to the rest of the primary connectives: the limited contraction | of Jäger (2005) has application to anaphora and the limited weakening W of Morrill and Valentín (2014) has application to words as types. The remaining semantically inactive connectives were intro-

[6] I thank Max Kanovich for drawing my attention to this design possibility.

duced as follows. Semantically inactive multiplicatives {•⊸, ⊸, ∘⊸, ⊸•, ◐, ◑, ⍏, ⍖, ⍐, ⍗, ◓, ◒}: Morrill and Valentín (2014). Semantically inactive additives {⊓, ⊔}: Morrill (1994). Semantically inactive first-order quantifiers {∀, ∃}: Morrill (1994). Semantically inactive normal modalities {■, ♦}: Hepple (1990b); Morrill (1994). The rules for semantically inactive variants of connectives are the same as those for the semantically active versions syntactically, but have invariant semantic labels on premises and conclusions.[7]

3 Focusing

Spurious ambiguity or derivational equivalence is the phenomenon whereby distinct analyses in grammar may assign the same derivational reading, resulting in redundancy in the parse search space and inefficiency in parsing. Understanding the problem depends on identifying the essential mathematical structure of derivations. This is trivial in the case of context free grammar, where the parse structures are ordered trees; in the case of logical categorial grammar, where the parse structures embody also semantic composition, these parse structures are proof nets. However, with respect to multiplicatives, fully intrinsic (structural) proof nets have not yet been given for displacement calculus — but see however Morrill and Fadda (2008); Fadda (2010); and Moot (2014, 2016).

Therefore, provisionally *CatLog* operates by Cut-free backward chaining sequent proof search and approaches spurious ambiguity by using Andreoli's proof-theoretic technique of focalization (Andreoli, 1992), which engenders a great reduction of derivational equivalence, although full canonicality requires so-called *multifocusing* (Chaudhuri et al., 2008). For another approach on the subject of unfocused and focused versions of the sequent calculus of an extended Lambek Grammar see Moortgat and Moot (2013).

Focusing is based on the distinction between rules which are invertible/reversible and rules which are non-invertible/irreversible. An example of a unary reversible rule is $/R$ since it is valid reading both from premise to conclusion and from conclusion to premise. An example of a unary irreversible rule is $\&L$. An example of a binary reversible rule is $\&R$ since it is valid not only reading from premises to conclusion, but also reading from conclusion to premises. An example of a binary irreversible rule is $/L$.

In proof search, a type of a sequent either eventually shows up outermost in an antecedent position or else in succedent position; we say that subtypes are *situated* accordingly: input (or $^\bullet$) for antecedent, and output (or $^\circ$) for succedent. In focalization (situated) types are classified as of *reversible/negative* or *irreversible/positive polarity* according to whether their associated rule is reversible or not.

In focused proof search, a sequent is either unfocused, in which case it is unboxed, as before, or else focused, and has exactly one type boxed. This is the focused type. In focusing there are alternating phases of don't-care nondeterministic negative rule application on the one hand, and positive rule application locking on to *focalized* formulas on the other hand. Given a sequent, zero or more invertible rules are applied in any fashion until there are no occurrences of negative formulas; then one chooses a positive formula as principal formula (which is boxed; it is focalized) and applies proof search to its subformulas while these remain positive. When one finds a negative formula, invertible rules are applied in a don't care nondeterministic fashion again until no longer possible, when another positive formula

[7] The synthetic connectives are: left and right projection and injection $\{\triangleleft^{-1}, \triangleright^{-1}, \triangleleft, \triangleright\}$, (Morrill et al., 2009); split and bridge {ˆ, ˆ}, (Morrill and Merenciano, 1996); and continuous and discontinuous nondeterministic multiplicatives {÷, ×, ⇑, ⇓, ⊚}, (Morrill et al., 2011).

$$\frac{\overrightarrow{A}: x, \Gamma \Rightarrow C: \chi}{\Gamma \Rightarrow A\backslash C: \lambda x \chi}\backslash R \qquad \frac{\Gamma, \overrightarrow{B}: y \Rightarrow C: \chi}{\Gamma \Rightarrow C/B: \lambda y \chi}/R$$

$$\frac{\Delta\langle \overrightarrow{A}: x, \overrightarrow{B}: y\rangle \Rightarrow D: \omega}{\Delta\langle \overrightarrow{A \bullet B}: z\rangle \Rightarrow D: \omega\{\pi_1 z/x, \pi_2 z/y\}}\bullet L$$

$$\frac{\Delta\langle \Lambda\rangle \Rightarrow A: \phi}{\Delta\langle \overrightarrow{I}: x\rangle \Rightarrow A: \phi} IL$$

$$\frac{\overrightarrow{A}: x \mid_k \Gamma \Rightarrow C: \chi}{\Gamma \Rightarrow A \downarrow_k C: \lambda x \chi}\downarrow_k R \qquad \frac{\Gamma \mid_k \overrightarrow{B}: y \Rightarrow C: \chi}{\Gamma \Rightarrow C \uparrow_k B: \lambda y \chi}\uparrow_k R$$

$$\frac{\Delta\langle \overrightarrow{A}: x \mid_k \overrightarrow{B}: y\rangle \Rightarrow D: \omega}{\Delta\langle \overrightarrow{A \odot_k B}: z\rangle \Rightarrow D: \omega\{\pi_1 z/x, \pi_2 z/y\}}\odot_k L$$

$$\frac{\Delta\langle 1\rangle \Rightarrow A: \phi}{\Delta\langle \overrightarrow{J}: x\rangle \Rightarrow A: \phi} JL$$

Fig. 8 Invertible multiplicative rules

$$\frac{\Gamma \Rightarrow A: \phi \qquad \Gamma \Rightarrow B: \psi}{\Gamma \Rightarrow A \& B: (\phi, \psi)}\&R$$

$$\frac{\Gamma\langle \overrightarrow{A}: x\rangle \Rightarrow C: \chi_1 \qquad \Gamma\langle \overrightarrow{B}: y\rangle \Rightarrow C: \chi_2}{\Gamma\langle \overrightarrow{A \oplus B}: z\rangle \Rightarrow C: z \to x.\chi_1; y.\chi_2}\oplus L$$

Fig. 9 Invertible additive rules

is chosen, and so on. *CatLog3* can be set to focus all atoms in the input (as in the examples later in this article) or in the output (i.e. it implements uniform *bias*).

The focalized logical rules for displacement calculus with additives are given in Figures 8–15. The focusing rules leading from a negative phase to a positive phase are:

$$\frac{\Gamma \Rightarrow \boxed{\mathrm{P}}}{\Gamma \Rightarrow P} FR$$

$$\frac{\Gamma\langle \boxed{\mathrm{Q}}\rangle \Rightarrow A}{\Gamma\langle Q\rangle \Rightarrow A} FL$$

The completness of such focusing for displacement calculus with additives is proved in Morrill and Valentín (2018). We suppress the focalized logical rules for connectives other than those of additive displacement calculus; the completeness of focalization for these other connectives of *CatLog3* is a topic of ongoing research.

$$\frac{\Gamma \Rightarrow \boxed{P} : \phi \qquad \Delta\langle \overrightarrow{\boxed{Q}} : z\rangle \Rightarrow D : \omega}{\Delta\langle \Gamma, \overrightarrow{\boxed{P \backslash Q}} : y\rangle \Rightarrow D : \omega\{(y\ \phi)/z\}}\ \backslash L$$

$$\frac{\Gamma \Rightarrow \boxed{P_1} : \phi \qquad \Delta\langle \overrightarrow{P_2} : z\rangle \Rightarrow D : \omega}{\Delta\langle \Gamma, \overrightarrow{\boxed{P_1 \backslash P_2}} : y\rangle \Rightarrow D : \omega\{(y\ \phi)/z\}}\ \backslash L$$

$$\frac{\Gamma \Rightarrow Q_1 : \phi \qquad \Delta\langle \overrightarrow{\boxed{Q_2}} : z\rangle \Rightarrow D : \omega}{\Delta\langle \Gamma, \overrightarrow{\boxed{Q_1 \backslash Q_2}} : y\rangle \Rightarrow D : \omega\{(y\ \phi)/z\}}\ \backslash L$$

$$\frac{\Gamma \Rightarrow Q : \phi \qquad \Delta\langle \overrightarrow{P} : z\rangle \Rightarrow D : \omega}{\Delta\langle \Gamma, \overrightarrow{\boxed{Q \backslash P}} : y\rangle \Rightarrow D : \omega\{(y\ \phi)/z\}}\ \backslash L$$

$$\frac{\Gamma \Rightarrow \boxed{P} : \psi \qquad \Delta\langle \overrightarrow{\boxed{Q}} : z\rangle \Rightarrow D : \omega}{\Delta\langle \overrightarrow{\boxed{Q / P}} : x, \Gamma\rangle \Rightarrow D : \omega\{(x\ \psi)/z\}}\ /L$$

$$\frac{\Gamma \Rightarrow Q_1 : \psi \qquad \Delta\langle \overrightarrow{\boxed{Q_2}} : z\rangle \Rightarrow D : \omega}{\Delta\langle \overrightarrow{\boxed{Q_2 / Q_1}} : x, \Gamma\rangle \Rightarrow D : \omega\{(x\ \psi)/z\}}\ /L$$

$$\frac{\Gamma \Rightarrow \boxed{P_1} : \psi \qquad \Delta\langle \overrightarrow{P_2} : z\rangle \Rightarrow D : \omega}{\Delta\langle \overrightarrow{\boxed{P_2 / P_1}} : x, \Gamma\rangle \Rightarrow D : \omega\{(x\ \psi)/z\}}\ /L$$

$$\frac{\Gamma \Rightarrow Q : \psi \qquad \Delta\langle \overrightarrow{P} : z\rangle \Rightarrow D : \omega}{\Delta\langle \overrightarrow{\boxed{P / Q}} : x, \Gamma\rangle \Rightarrow D : \omega\{(x\ \psi)/z\}}\ /L$$

Fig. 10 Left noninvertible continuous multiplicative rules

$$\frac{\Gamma \Rightarrow \boxed{P} : \phi \qquad \Delta\langle \overrightarrow{\boxed{Q}} : z \rangle \Rightarrow D : \omega}{\Delta\langle \Gamma \mid_k \overrightarrow{\boxed{P \downarrow_k Q}} : y \rangle \Rightarrow D : \omega\{(y\ \phi)/z\}} \downarrow_k L$$

$$\frac{\Gamma \Rightarrow \boxed{P_1} : \phi \qquad \Delta\langle \overrightarrow{P_2} : z \rangle \Rightarrow D : \omega}{\Delta\langle \Gamma \mid_k \overrightarrow{\boxed{P_1 \downarrow_k P_2}} : y \rangle \Rightarrow D : \omega\{(y\ \phi)/z\}} \downarrow_k L$$

$$\frac{\Gamma \Rightarrow Q_1 : \phi \qquad \Delta\langle \overrightarrow{\boxed{Q_2}} : z \rangle \Rightarrow D : \omega}{\Delta\langle \Gamma \mid_k \overrightarrow{\boxed{Q_1 \downarrow_k Q_2}} : y \rangle \Rightarrow D : \omega\{(y\ \phi)/z\}} \downarrow_k L$$

$$\frac{\Gamma \Rightarrow Q : \phi \qquad \Delta\langle \overrightarrow{P} : z \rangle \Rightarrow D : \omega}{\Delta\langle \Gamma \mid_k \overrightarrow{\boxed{Q \downarrow_k P}} : y \rangle \Rightarrow D : \omega\{(y\ \phi)/z\}} \downarrow_k L$$

$$\frac{\Gamma \Rightarrow \boxed{P} : \psi \qquad \Delta\langle \overrightarrow{\boxed{Q}} : z \rangle \Rightarrow D : \omega}{\Delta\langle \overrightarrow{\boxed{Q \uparrow_k P}} : x \mid_k \Gamma \rangle \Rightarrow D : \omega\{(x\ \psi)/z\}} \uparrow_k L$$

$$\frac{\Gamma \Rightarrow Q_1 : \psi \qquad \Delta\langle \overrightarrow{\boxed{Q_2}} : z \rangle \Rightarrow D : \omega}{\Delta\langle \overrightarrow{\boxed{Q_2 \uparrow_k Q_1}} : x \mid_k \Gamma \rangle \Rightarrow D : \omega\{(x\ \psi)/z\}} \uparrow_k L$$

$$\frac{\Gamma \Rightarrow \boxed{P_1} : \psi \qquad \Delta\langle \overrightarrow{P_2} : z \rangle \Rightarrow D : \omega}{\Delta\langle \overrightarrow{\boxed{P_2 \uparrow_k P_1}} : x \mid_k \Gamma \rangle \Rightarrow D : \omega\{(x\ \psi)/z\}} \uparrow_k L$$

$$\frac{\Gamma \Rightarrow Q : \psi \qquad \Delta\langle \overrightarrow{P} : z \rangle \Rightarrow D : \omega}{\Delta\langle \overrightarrow{\boxed{P \uparrow_k Q}} : x \mid_k \Gamma \rangle \Rightarrow D : \omega\{(x\ \psi)/z\}} \uparrow_k L$$

Fig. 11 Left noninvertible discontinuous multiplicative rules

$$\frac{\Gamma\langle \overrightarrow{\boxed{Q}} : x\rangle \Rightarrow C : \chi}{\Gamma\langle \overrightarrow{\boxed{Q\&B}} : z\rangle \Rightarrow C : \chi\{\pi_1 z/x\}}\ \&L_1$$

$$\frac{\Gamma\langle \overrightarrow{P} : x\rangle \Rightarrow C : \chi}{\Gamma\langle \overrightarrow{\boxed{P\&B}} : z\rangle \Rightarrow C : \chi\{\pi_1 z/x\}}\ \&L_1$$

$$\frac{\Gamma\langle \overrightarrow{\boxed{Q}} : y\rangle \Rightarrow C : \chi}{\Gamma\langle \overrightarrow{\boxed{A\&Q}} : z\rangle \Rightarrow C : \chi\{\pi_2 z/y\}}\ \&L_2$$

$$\frac{\Gamma\langle \overrightarrow{P} : y\rangle \Rightarrow C : \chi}{\Gamma\langle \overrightarrow{\boxed{A\&P}} : z\rangle \Rightarrow C : \chi\{\pi_2 z/y\}}\ \&L_2$$

Fig. 12 Left noninvertible additive rules

$$\frac{\Delta \Rightarrow \boxed{P_1} : \phi \qquad \Gamma \Rightarrow \boxed{P_2} : \psi}{\Delta, \Gamma \Rightarrow \boxed{P_1 \bullet P_2} : (\phi, \psi)}\ \bullet R$$

$$\frac{\Delta \Rightarrow \boxed{P} : \phi \qquad \Gamma \Rightarrow Q : \psi}{\Delta, \Gamma \Rightarrow \boxed{P \bullet Q} : (\phi, \psi)}\ \bullet R$$

$$\frac{\Delta \Rightarrow Q : \phi \qquad \Gamma \Rightarrow \boxed{P} : \psi}{\Delta, \Gamma \Rightarrow \boxed{Q \bullet P} : (\phi, \psi)}\ \bullet R$$

$$\frac{\Delta \Rightarrow Q_1 : \phi \qquad \Gamma \Rightarrow Q_2 : \psi}{\Delta, \Gamma \Rightarrow \boxed{Q_1 \bullet Q_2} : (\phi, \psi)}\ \bullet R$$

$$\frac{}{\Lambda \Rightarrow \boxed{I} : 0}\ IR$$

Fig. 13 Right noninvertible continuous multiplicative rules

$$\frac{\Delta \Rightarrow \boxed{P_1} : \phi \qquad \Gamma \Rightarrow \boxed{P_2} : \psi}{\Delta \mid_k \Gamma \Rightarrow \boxed{P_1 \odot_k P_2} : (\phi, \psi)} \odot_k R$$

$$\frac{\Delta \Rightarrow \boxed{P} : \phi \qquad \Gamma \Rightarrow Q : \psi}{\Delta \mid_k \Gamma \Rightarrow \boxed{P \odot_k Q} : (\phi, \psi)} \odot_k R$$

$$\frac{\Delta \Rightarrow Q : \phi \qquad \Gamma \Rightarrow \boxed{P} : \psi}{\Delta \mid_k \Gamma \Rightarrow \boxed{Q \odot_k P} : (\phi, \psi)} \odot_k R$$

$$\frac{\Delta \Rightarrow Q_1 : \phi \qquad \Gamma \Rightarrow Q_2 : \psi}{\Delta \mid_k \Gamma \Rightarrow \boxed{Q_1 \odot_k Q_2} : (\phi, \psi)} \odot_k R$$

$$\frac{}{1 \Rightarrow \boxed{J} : 0} JR$$

Fig. 14 Right noninvertible discontinuous multiplicative rules

$$\frac{\Gamma \Rightarrow \boxed{P} : \phi}{\Gamma \Rightarrow \boxed{P \oplus B} : \iota_1 \phi} \oplus R_1 \qquad \frac{\Gamma \Rightarrow Q : \phi}{\Gamma \Rightarrow \boxed{Q \oplus B} : \iota_1 \phi} \oplus R_1$$

$$\frac{\Gamma \Rightarrow \boxed{P} : \psi}{\Gamma \Rightarrow \boxed{A \oplus P} : \iota_2 \psi} \oplus R_2 \qquad \frac{\Gamma \Rightarrow Q : \psi}{\Gamma \Rightarrow \boxed{A \oplus Q} : \iota_2 \psi} \oplus R_2$$

Fig. 15 Right noninvertible additive rules

This system, which is Cut-free, is what we call *strongly* focalized. Pseudo-code for the *CatLog3* parsing/theorem-proving algorithm of Cut-free backward chaining strongly focalized h-sequent proof search is as follows:

```
function prove(Σ: unfocused sequent): bool;

/* prove(Σ) returns true if the unfocused sequent Σ is provable; otherwise it returns
false. */

return prove_rev_lst([Σ]).

function prove_rev_lst(Ls: list of unfocused sequents): bool;

/* prove_rev_lst(Ls) returns true if the list Ls of unfocused sequents are provable;
otherwise it returns false. */

while Ls contains a sequent Σ with a reversible type do
    Ls := Ls with Σ replaced by the premises of the rule for the reversible type;
return prove_irrev_lst(Ls).

function prove_irrev_lst(Ls: list of unfocused sequents): bool;

/* prove_irrev_lst(Ls) returns true if the list Ls of unfocused sequents without reversible
types are provable; otherwise it returns false. */

var Success: bool;
if Ls = [] then return true
    else where Ls = [Σ|Ls'] do
        begin
        Success := false;
        for each irreversible type in Σ do
            begin
            focus (i.e. box) in Σ this type, obtaining Σ';
            if prove_irrev(Σ') then Success := true;
            end;
        return Success and prove_irrev_lst(Ls');
        end.

function prove_irrev(Σ: focused sequent): bool;

/* prove_irrev(Σ) returns true if the focused sequent Σ is provable; otherwise it returns
false. */

var Success: bool;
var Ls_rev: list of unfocused sequents;
var Ls_irrev: list of focused sequents;
Success := false;
for each rule application to the focused type in Σ do
    begin
    Ls_rev := its reversible premises;
    Ls_irrev := its irreversible premises;
    if prove_rev_lst(Ls_rev) and prove_irrev_lst(Ls_irrev) then Success := true
    end;
return Success.
```

The exhaustive semantic parsing of *CatLog3* requires lexical lookup and exhaustive computation of the readings of distinct derivations. The algorithm just given computes only the characteristic function of derivability, i.e. whether a sequent is a theorem or not, but it is exhaustive like *CatLog3* in that it explores the whole derivation search space. The

top level call to determine whether a sequent Σ is provable is `prove(`Σ`)`. The routine `prove(`S`)` calls the routine `prove_rev_lst` with parameter the unitary list `[`S`]`. The routine `prove_rev_lst` then applies reversible rules to its list of sequents `Ls` in a don't-care nondeterministic manner until none of the sequents contain any reversible type, i.e. it closes `Ls` under reversible rules. Then, `prove_irrev_lst` is called on the resulting list of irreversible sequents. This calls `prove_irrev(`Σ'`)` for focusings Σ' of each sequent, and if some focusing of each sequent is provable the result **true** is returned; otherwise **false** is returned. The procedure `prove_irrev` applies focusing rules and recurses back on `prove_rev_lst` and `prove_irrev_lst` to determine provability for the given focusings.

4 Count-invariance

Count-invariance affords a simply checked condition which is necessary though not sufficient for provability. In backward chaining proof search each new subgoal generated can be rapidly checked for the count-invariance property and, if it does not satisfy it, be immediately discarded without further ado. This affords effective pruning of proof search in categorial parsing/theorem-proving. We define infinitary inequational count-invariance for categorial logic, extending the equational count-invariance for multiplicatives of van Benthem (1991) and the inequational count-invariance for additives of Valentín et al. (2013) to include exponentials (Kuznetsov et al., 2017).

The count-invariance for multiplicatives in sublinear logic introduced by van Benthem just involves checking the number of positive and negative occurrences of each atom in a sequent. Thus where $\#(\Sigma)$ is a count of the sequent Σ we have:

(5) $\vdash \Sigma \Longrightarrow \#(\Sigma) = 0$.

I.e. the numbers of positive and negative occurrences of each atom must exactly balance. This provides a necessary, but of course not sufficient, criterion for theoremhood of multiplicative sequents, and it can be checked rapidly. It can be used as a filter in proof search: if backward chaining proof search generates a goal which does not satisfy the count invariant, the goal can be safely made to fail immediately. This notion of count for multiplicatives was included in the categorial parser/theorem-prover *CatLog1*.

In Valentín et al. (2013) the idea is extended to additives. For this, instead of a single count for each atom of a sequent Σ we have a minimum count $\#_{\min}(\Sigma)$ and a maximum count $\#_{\max}(\Sigma)$ and for a sequent to be a theorem it must satisfy two inequations:

(6) $\vdash \Sigma \Longrightarrow \#_{\min}(\Sigma) \leq 0 \leq \#_{\max}(\Sigma)$.

I.e. the count functions $\#_{\min}$ and $\#_{\max}$ must define an interval which includes the point of exact balance 0; for multiplicative sequents, $\#_{\min} = \#_{\max} = \#$ and (6) reduces to the special case (5). This count-invariance is included in the categorial parser/theorem-prover *CatLog2*.

Here we describe the count-invariance of *CatLog3* which includes more general, infinitary, count functions, for exponentials (Kuznetsov et al., 2017). We consider terms built over the constants 0, 1, $\bot$ (minus infinity, $-\infty$), and $\top$ (plus infinity, $+\infty$) by operations plus (+), minus (−), minimum (min) and maximum (max), and infinitary step functions X and Y thus where $i, j \in \mathcal{Z}$:[8]

[8] Undefined values in infinitary arithmetic are indicated by $*$; we could have opted to fail to reject any counts which are undefined, but in fact such cases do not ever occur (Kuznetsov et al., 2017).

$$
\begin{array}{rcll}
\#^p_{m,P}(Q) & = & \begin{array}{l} 1 \text{ if } Q = P \\ 0 \text{ if } Q \neq P \end{array} & \\
\#^p_{m,P}(A\backslash C) & = & \#^p_{m,Q}(A{\downarrow_k}C) & = \#^p_{m,P}(C) - \#^{\overline{p}}_{\overline{m},P}(A) \\
\#^p_{m,P}(C/B) & = & \#^p_{m,P}(C{\uparrow_k}B) & = \#^p_{m,P}(C) - \#^{\overline{p}}_{\overline{m},P}(B) \\
\#^p_{m,P}(A\bullet B) & = & \#^p_{m,P}(A{\odot_k}B) & = \#^p_{m,P}(A) + \#^p_{m,P}(B) \\
\#^p_{m,P}(I) & = & \#^p_{m,P}(J) & = 0 \\
\#^{\circ}_{m,P}(A\&B) & = & \overline{m}(\#^{\circ}_{m,P}(A), \#^{\circ}_{m,P}(B)) & \\
\#^{\bullet}_{m,P}(A\&B) & = & m(\#^{\bullet}_{m,P}(A), \#^{\bullet}_{m,P}(B)) & \\
\#^{\circ}_{m,P}(A\oplus B) & = & m(\#^{\circ}_{m,P}(A), \#^{\circ}_{m,P}(B)) & \\
\#^{\bullet}_{m,P}(A\oplus B) & = & \overline{m}(\#^{\bullet}_{m,P}(A), \#^{\bullet}_{m,P}(B)) & \\
\#^p_{m,P}(\wedge\, xA) & = & \#^p_{m,P}(\vee\, xA) & = \#^p_{m,P}(A) \\
\#^p_{m,P}(\Box A) & = & \#^p_{m,P}(\Diamond A) & = \#^p_{m,P}(A) \\
\#^p_{m,P}([\,]^{-1}A) & = & \#^p_{m,P}(A) & \\
\#^p_{m,P}(\langle\rangle A) & = & \#^p_{m,P}(A) & \\
\#^{\bullet}_{\min,P}(!A) & = & Y(\#^{\bullet}_{\min,P}(A)) & \\
\#^{\bullet}_{\max,P}(!A) & = & X(\#^{\bullet}_{\max,P}(A)) & \\
\#^{\circ}_{m,P}(!A) & = & \#^{\circ}_{m,P}(A) & \\
\#^{\circ}_{\min,P}(?A) & = & Y(\#^{\circ}_{\min,P}(A)) & \\
\#^{\circ}_{\max,P}(?A) & = & X(\#^{\circ}_{\max,P}(A)) & \\
\#^{\bullet}_{m,P}(?A) & = & \#^{\bullet}_{m,P}(A) &
\end{array}
$$

Fig. 16 Count function

$+$	j	$\bot$	$\top$
i	$i{+}j$	$\bot$	$\top$
$\bot$	$\bot$	$\bot$	$*$
$\top$	$\top$	$*$	$\top$

$-$	j	$\bot$	$\top$
i	$i{-}j$	$\top$	$\bot$
$\bot$	$\bot$	$*$	$\bot$
$\top$	$\top$	$\top$	$*$

min	j	$\bot$	$\top$
i	$min(i,j)$	$\bot$	i
$\bot$	$\bot$	$\bot$	$\bot$
$\top$	j	$\bot$	$\top$

max	j	$\bot$	$\top$
i	$max(i,j)$	i	$\top$
$\bot$	j	$\bot$	$\top$
$\top$	$\top$	$\top$	$\top$

$$X(i) = \begin{cases} \top \text{ if } i > 0 \\ i \text{ if } i \leq 0 \end{cases} \qquad Y(i) = \begin{cases} i \text{ if } i \geq 0 \\ \bot \text{ if } i < 0 \end{cases}$$

Where $\mathcal{P}$ is the set of primitive types, $P \in \mathcal{P}$;

- $p \in \{\bullet, \circ\}$, $\overline{\bullet} = \circ$, $\overline{\circ} = \bullet$;
- $m \in \{\min, \max\}$, $\overline{\min} = \max$, $\overline{\max} = \min$;

we define the count functions for our fragment of displacement logic as shown in Figure 16. For zones, stoups, configurations and tree terms, counts are as follows:

$$\begin{aligned}
\#^{\bullet}{}_{m,P}(\mathcal{S};\mathcal{O}) &= \#^{\bullet}{}_{m,P}(\mathcal{S}) + \#^{\bullet}{}_{m,P}(\mathcal{O})\\
\#^{\bullet}{}_{m,P}(\emptyset) &= 0\\
\#^{\bullet}{}_{m,P}(\mathcal{F},\mathcal{S}) &= \#^{\bullet}{}_{m,P}(\mathcal{F}) + \#^{\bullet}{}_{m,P}(\mathcal{S})\\
\#^{\bullet}{}_{m,P}(\Lambda) &= 0\\
\#^{\bullet}{}_{m,P}(\mathcal{T},\mathcal{O}) &= \#^{\bullet}{}_{m,P}(\mathcal{T}) + \#^{\bullet}{}_{m,P}(\mathcal{O})\\
\#^{\bullet}{}_{m,P}(1) &= 0\\
\#^{\bullet}{}_{m,P}(\mathcal{F}) &= \#^{\bullet}{}_{m,P}(\mathcal{F})\\
\#^{\bullet}{}_{m,P}(\mathcal{F}\{\mathcal{O}_1 : \ldots : \mathcal{O}_i\}) &= \#^{\bullet}{}_{m,P}(\mathcal{F}) + \Sigma_{n=1}^{i}\#^{\bullet}{}_{m,P}(\mathcal{O}_n)\\
\#^{\bullet}{}_{m,P}([\mathcal{Z}]) &= \#^{\bullet}{}_{m,P}(\mathcal{Z})
\end{aligned}$$

The count-invariance theorem is:

(7) **Theorem**.
$\vdash \Xi \Rightarrow A \Longrightarrow \forall P \in \mathcal{P}, \#_{\min,P}(\Xi \Rightarrow A) \leq 0 \leq \#_{\max,P}(\Xi \Rightarrow A)$,
where $\#_{m,P}(\Xi \Rightarrow A) = \#^{\circ}_{m,P}(A) - \#^{\bullet}_{\overline{m},P}(\Xi)$.

We have for instance that relativisation including medial and parasitic extraction is obtained by assigning a relative pronoun a categorial type $(CN\backslash CN)/(!N\backslash S)$ whereby the body of a relative clause is analysed as $!N\backslash S$. By way of example of count-invariance for !, we show how it discards $N, N\backslash S \Rightarrow !N\backslash S$ corresponding to the ungrammaticality of a relative clause without a gap: **paper that John walks*. We have the max N-count: $\#_{\max,N}(N, N\backslash S \Rightarrow !N\backslash S) = \#^{\circ}_{\max,N}(!N\backslash S) - \#^{\bullet}_{\min,N}(N, N\backslash S) = \#^{\circ}_{\max,N}(S) - \#^{\bullet}_{\min,N}(!N) - \#^{\bullet}_{\min,N}(N) - \#^{\bullet}_{\min,N}(N\backslash S) = 0 - Y(\#^{\bullet}_{\min,N}(N)) - 1 - (\#^{\bullet}_{\min,N}(S) - \#^{\circ}_{\min,N}(N)) = -Y(1) - 1 - (0-1) = -1 - 1 + 1 = -1 \not\geq 0$ which means that the count-invariance is not satisfied.

By way of a second example, we have that iterated sentential coordination is obtained by assigning a coordinator the type $(?S\backslash S)/S$. Illustrating count-invariance for ?, we show how it discards $N, N, N\backslash S \Rightarrow ?S$ corresponding to the ungrammaticality of unequilibrated coordination: **John Mary walks and Suzy talks*. For this example max N-count is: $\#_{\max,N}(N, N, N\backslash S \Rightarrow ?S) = \#^{\circ}_{\max,N}(?S) - \#^{\bullet}_{\min,N}(N, N, N\backslash S) = X(\#^{\circ}_{\max,N}(S)) - \#^{\bullet}_{\min,N}(N) - \#^{\bullet}_{\min,N}(N) - \#^{\bullet}_{\min,N}(N\backslash S) = X(0) - 1 - 1 - (\#^{\bullet}_{\min,N}(S) - \#^{\bullet}_{\max,N}(N)) = 0 - 2 - 0 + 1 = -1 \not\geq 0$ which means that the count-invariance is not satisfied.

5 Narrow-scale evaluation

Both *CatLog2* and *CatLog3* use focusing and both multiplicative and additive count-invariance, but only the latter has subexponential count-invariance; by way of evaluation of this difference, we compared the performance of *CatLog2* (version f8.1) with *CatLog3* (version j2); apart from the exponential count-invariance the engines were the same, both running under XGP Prolog on a MacBook Air, and with identical lexicons and example sentences. We timed individually the exhaustive parsing of the expressions in Figure 17. The results, in seconds, were as follows:

a. John likes the man.
b. Mary thinks that John likes the man.
c. Suzy believes that Mary thinks that John likes the man.
d. man that John likes
e. man that Mary thinks that John likes
f. man that Suzy believes that Mary thinks that John likes
g. Mary talks and Bill sings.
h. John walks Mary talks and Bill sings.
i. Suzy laughs John walks Mary talks and Bill sings.
j. Bill walks Suzy laughs John walks Mary talks and Bill sings.
k. Suzy talks Bill walks Suzy laughs John walks Mary talks and Bill sings.
l. John sings Suzy talks Bill walks Suzy laughs John walks Mary talks and Bill sings.

Fig. 17 Example sentences

(8)

	CatLog2 (version f8.1)	*CatLog3* (version j2)
a.	1	1
b.	2	2
c.	40	6
d.	2	2
e.	4	4
f.	265	6
g.	1	1
h.	2	1
i.	2	1
j.	2	2
k.	2	3
l.	2	4

The canonical and non-canonical examples a-c and d-f depend on *that* which is lexically ambiguous between a complementiser, and a relative pronoun roughly of the form $(CN\backslash CN)/((\langle\rangle N \sqcap !N)\backslash S)$ (where the semantically inactively additively conjoined hypothetical subtypes are for subject relativisation and object relativisation), respectively. We see that for the longer, c, example there is a considerable speedup. This would appear to reflect the time required by *CatLog2* for dismissal, without the benefit of exponential count-invariance, of inappropriate lexical choices of the relative pronoun option. Likewise for the longer, f, example there is a considerable speedup. This would appear to reflect the time required by *CatLog2* for exhaustive perusal of the choice involving the universal exponential relative pronoun option. The examples g-l involve the existential exponential in a coordinator assignment roughly of the form $(?S\backslash[\,]^{-1}[\,]^{-1}S)/S$ to obtain the iteration. Here we see that there is no gain from the exponential type invariance; indeed with the longest examples, k-l, we find that the overhead even causes a slight slowdown; so in the next section we perform a wider, averaged, comparison.

6 Illustration and wider-scale evaluation

Morrill and Valentín (2016) defines as the *Montague Test* the task of providing a computational cover grammar of the PTQ fragment of Montague (1973), and shows how *CatLog* fulfils this task; we are not aware of any other system which has passed the Montague Test.

```
str(dwp('(7-7)'), [b([john]), walks], s(f)).
str(dwp('(7-16)'), [b([every, man]), talks], s(f)).
str(dwp('(7-19)'), [b([the, fish]), walks], s(f)).
str(dwp('(7-32)'), [b([every, man]), b([b([walks, or, talks])])], s(f)).
str(dwp('(7-34)'), [b([b([b([every, man]), walks, or, b([every, man]), talks])])], s(f)).
str(dwp('(7-39)'), [b([b([b([a, woman]), walks, and, b([she]), talks])])], s(f)).
str(dwp('(7-43, 45)'), [b([john]), believes, that, b([a, fish]), walks], s(f)).
str(dwp('(7-48, 49, 52)'), [b([every, man]), believes, that, b([a, fish]), walks], s(f)).
str(dwp('(7-57)'), [b([every, fish, such, that, b([it]), walks]), talks], s(f)).
str(dwp('(7-60, 62)'), [b([john]), seeks, a, unicorn], s(f)).
str(dwp('(7-73)'), [b([john]), is, bill], s(f)).
str(dwp('(7-76)'), [b([john]), is, a, man], s(f)).
str(dwp('(7-83)'), [necessarily, b([john]), walks], s(f)).
str(dwp('(7-86)'), [b([john]), walks, slowly], s(f)).
str(dwp('(7-91)'), [b([john]), tries, to, walk], s(f)).
str(dwp('(7-94)'), [b([john]), tries, to, b([b([catch, a, fish, and, eat, it])])], s(f)).
str(dwp('(7-98)'), [b([john]), finds, a, unicorn], s(f)).
str(dwp('(7-105)'), [b([every, man, such, that, b([he]), loves, a, woman]), loses, her], s(f)).
str(dwp('(7-110)'), [b([john]), walks, in, a, park], s(f)).
str(dwp('(7-116, 118)'), [b([every, man]), doesnt, walk], s(f)).
```

Fig. 18 Montague sentences

The example sentences of this test, those analysed in Chapter 7 of Dowty et al. (1981), are given in Figure 18; the lexicon is given in Figure 19.

a : $\blacksquare\forall g(\forall f((S\,f\uparrow\blacksquare Nt(s(g)))\downarrow S\,f)/CNs(g)) : \lambda A\lambda B\exists C[(A\ C)\wedge(B\ C)]$
and : $\blacksquare\forall f((\blacksquare ?S\,f\backslash[]^{-1}[]^{-1}S\,f)/\blacksquare S\,f) : (\Phi^{n+}\ 0\ and)$
and : $\blacksquare\forall a\forall f((\blacksquare ?(\langle\rangle Na\backslash S\,f)\backslash[]^{-1}[]^{-1}(\langle\rangle Na\backslash S\,f))/\blacksquare(\langle\rangle Na\backslash S\,f)) : (\Phi^{n+}\ (s\ 0)\ and)$
believes : $\square(((\langle\rangle\exists gNt(s(g))\backslash S\,f)/(CPthat\sqcup\square S\,f)) : \hat{}\lambda A\lambda B(Pres\ ((\check{}believe\ A)\ B))$
bill : $\blacksquare Nt(s(m)) : b$
catch : $\square(((\langle\rangle\exists aNa\backslash S\,b)/\exists aNa) : \hat{}\lambda A\lambda B((\check{}catch\ A)\ B)$
doesnt : $\blacksquare\forall g\forall a((S\,g\uparrow(((\langle\rangle Na\backslash S\,f)/(\langle\rangle Na\backslash S\,b)))\downarrow S\,g) : \lambda A\neg(A\ \lambda B\lambda C(B\ C))$
eat : $\square(((\langle\rangle\exists aNa\backslash S\,b)/\exists aNa) : \hat{}\lambda A\lambda B((\check{}eat\ A)\ B)$
every : $\blacksquare\forall g(\forall f((S\,f\uparrow Nt(s(g)))\downarrow S\,f)/CNs(g)) : \lambda A\lambda B\forall C[(A\ C)\rightarrow(B\ C)]$
finds : $\square(((\langle\rangle\exists gNt(s(g))\backslash S\,f)/\exists aNa) : \hat{}\lambda A\lambda B(Pres\ ((\check{}find\ A)\ B))$
fish : $\square CNs(n) : fish$
he : $\blacksquare[]^{-1}\forall g((\blacksquare S\,g|\blacksquare Nt(s(m)))/(\langle\rangle Nt(s(m))\backslash S\,g)) : \lambda AA$
her : $\blacksquare\forall g\forall a((((\langle\rangle Na\backslash S\,g)\uparrow\blacksquare Nt(s(f)))\downarrow(\blacksquare(\langle\rangle Na\backslash S\,g)|\blacksquare Nt(s(f)))) : \lambda AA$
in : $\square(\forall a\forall f((\langle\rangle Na\backslash S\,f)\backslash(\langle\rangle Na\backslash S\,f))/\exists aNa) : \hat{}\lambda A\lambda B\lambda C((\check{}in\ A)\ (B\ C))$
is : $\blacksquare(((\langle\rangle\exists gNt(s(g))\backslash S\,f)/(\exists aNa\oplus(\exists g((CNg/CNg)\sqcup(CNg\backslash CNg))-I))) :$
$\lambda A\lambda B(Pres\ (A\rightarrow C.[B=C];D.((D\ \lambda E[E=B])\ B)))$
it : $\blacksquare\forall f\forall a((((\langle\rangle Na\backslash S\,f)\uparrow\blacksquare Nt(s(n)))\downarrow(\blacksquare(\langle\rangle Na\backslash S\,f)|\blacksquare Nt(s(n)))) : \lambda AA$
it : $\blacksquare[]^{-1}\forall f((\blacksquare S\,f|\blacksquare Nt(s(n)))/(\langle\rangle Nt(s(n))\backslash S\,f)) : \lambda AA$
john : $\blacksquare Nt(s(m)) : j$
loses : $\square(((\langle\rangle\exists gNt(s(g))\backslash S\,f)/\exists aNa) : \hat{}\lambda A\lambda B(Pres\ ((\check{}lose\ A)\ B))$
loves : $\square(((\langle\rangle\exists gNt(s(g))\backslash S\,f)/\exists aNa) : \hat{}\lambda A\lambda B(Pres\ ((\check{}love\ A)\ B))$
man : $\square CNs(m) : man$
necessarily : $\blacksquare(S\,A/\square S\,A) : Nec$
or : $\blacksquare\forall f((\blacksquare ?S\,f\backslash[]^{-1}[]^{-1}S\,f)/\blacksquare S\,f) : (\Phi^{n+}\ 0\ or)$
or : $\blacksquare\forall a\forall f((\blacksquare ?(\langle\rangle Na\backslash S\,f)\backslash[]^{-1}[]^{-1}(\langle\rangle Na\backslash S\,f))/\blacksquare(\langle\rangle Na\backslash S\,f)) : (\Phi^{n+}\ (s\ 0)\ or)$
or : $\blacksquare\forall f((\blacksquare ?(S\,f/(\langle\rangle\exists gNt(s(g))\backslash S\,f))\backslash[]^{-1}[]^{-1}(S\,f/(\langle\rangle\exists gNt(s(g))\backslash S\,f)))/\blacksquare(S\,f/(\langle\rangle\exists gNt(s(g))\backslash S\,f))) :$
$(\Phi^{n+}\ (s\ 0)\ or)$
park : $\square CNs(n) : park$
seeks : $\square(((\langle\rangle\exists gNt(s(g))\backslash S\,f)/\square\forall a\forall f(((Na\backslash S\,f)/\exists bNb)\backslash(Na\backslash S\,f))) :$
$\hat{}\lambda A\lambda B((\check{}tries\ \hat{}((\check{}A\ \check{}find)\ B))\ B)$
she : $\blacksquare[]^{-1}\forall g((\blacksquare S\,g|\blacksquare Nt(s(f)))/(\langle\rangle Nt(s(f))\backslash S\,g)) : \lambda AA$
slowly : $\square\forall a\forall f(\square(\langle\rangle Na\backslash S\,f)\backslash(\langle\rangle\square Na\backslash S\,f)) : \hat{}\lambda A\lambda B(\check{}slowly\ \hat{}(\check{}A\ \check{}B))$
such+that : $\blacksquare\forall n((CNn\backslash CNn)/(S\,f|\blacksquare Nt(n))) : \lambda A\lambda B\lambda C[(B\ C)\wedge(A\ C)]$
talks : $\square(\langle\rangle\exists gNt(s(g))\backslash S\,f) : \hat{}\lambda A(Pres\ (\check{}talk\ A))$
that : $\blacksquare(CPthat/\square S\,f) : \lambda AA$
the : $\blacksquare\forall n(Nt(n)/CNn) : \iota$
to : $\blacksquare((PPto/\exists aNa)\sqcap\forall n(((\langle\rangle Nn\backslash S\,i)/(\langle\rangle Nn\backslash S\,b))) : \lambda AA$
tries : $\square(((\langle\rangle\exists gNt(s(g))\backslash S\,f)/\square(\langle\rangle\exists gNt(s(g))\backslash S\,i)) : \hat{}\lambda A\lambda B((\check{}tries\ \hat{}(\check{}A\ B))\ B)$
unicorn : $\square CNs(n) : unicorn$
walk : $\square(\langle\rangle\exists aNa\backslash S\,b) : \hat{}\lambda A(\check{}walk\ A)$
walks : $\square(\langle\rangle\exists gNt(s(g))\backslash S\,f) : \hat{}\lambda A(Pres\ (\check{}walk\ A))$
woman : $\square CNs(f) : woman$

Fig. 19 Montague lexicon

CatLog3 generates the linguistic examples invoked, the results of their lexical lookup, their derivations, and their normalized semantic readings. It outputs this both in text, to the Prolog console, and in LaTeX, to files. The *CatLog3* LaTeX output, formatted into annotated displays and figures, for the (ambiguous) last Montague sentence is as follows.

The linguistic example is:

(9) (dwp((7-116, 118))) [**every**+**man**]+**doesnt**+**walk** : $S\,f$

The result of lexical lookup is:

(10) [$\blacksquare\forall g(\forall f((S\,f\uparrow Nt(s(g)))\downarrow S\,f)/CNs(g)) : \lambda A\lambda B\forall C[(A\ C)\rightarrow(B\ C)]$,
$\square CNs(m) : man$], $\blacksquare\forall g\forall a((S\,g\uparrow(((\langle\rangle Na\backslash S\,f)/(\langle\rangle Na\backslash S\,b)))\downarrow S\,g) : \lambda D\neg(D\ \lambda E\lambda F(E\ F))$,
$\square(\langle\rangle\exists aNa\backslash S\,b) : \hat{}\lambda G(\check{}walk\ G)\ \Rightarrow\ S\,f$

The $\forall > \neg$ derivation is given in Figure 10. This yields logical form:

(11) $\forall C[(\check{}man\ C) \rightarrow \neg(\check{}walk\ C)]$

The $\neg > \forall$ derivation is that given in Figure 21. This delivers logical form:

(12) $\neg\forall G[(\check{}man\ G) \rightarrow (\check{}walk\ G)]$

We compared performance of *CatLog2* and *CatLog3* for both the Montague minicorpus and the entire *CatLog2* corpus (Montague minicorpus, typical categorial examples, discontinuity examples, relativisation examples, coordination examples, and some Scripture). The times in seconds of *CatLog2* and *CatLog3* exhaustive parsing were as follows:

(13)

	CatLog2 (version f8.1)	*CatLog3* (version j2)
Montague Test	37	32
CatLog2 corpus	826	643

This indicates that overall the pruning of the search space that the *CatLog3* exponential count invariance engenders outweighs its processing cost overhead, delivering over the whole *CatLog2* corpus of examples an average speedup of around 20%.

7 Conclusion

CatLog originated around 1990 as a Prolog Cut-free backward chaining sequent proof search semantic parser/theorem-prover for logical categorial grammar. Its categorial fragment was an extension of Lambek calculus with connectives such as additives and normal modalities. It's approach to spurious ambiguity was guided by König (1989), Hepple (1990a) and Hendriks (1993) — in effect uniform proof for the Lambek calculus. The main challenge faced was to extend the fragment to discontinuity by means of multimodality. This approach was abandoned in 1995 because the search space induced by the multimodal structural postulates created prohibitive inefficiency.

The subsequent 15 years saw many intermittent advances on discontinuity but computationally it was not until 2010 when Oriol Valentín had the idea of h-sequent calculus that *CatLog* could be resumed including discontinuity without the prohibitive computational cost of structural postulates; this was done on the basis of uniform proof and multiplicative count-invariance. Uniform proof, however, is essentially for only a logic programming fragment; and van Benthem's count-invariance only works for additive- and exponential-free types. The approach to spurious ambiguity of *CatLog* was therefore subsequently switched to focusing; and *CatLog* was extended with additive count-invariance (*CatLog2*) and exponential count-invariance (*CatLog3*).

Although *CatLog3* parses examples which have no bracket domains, when bracket domains occur in the linguistic input they are required to be placed there. To truly parse all cases (without ever having to put brackets in the input) we want the proof search to induce/discover the appropriate bracketing. A first study in this respect, which is for the Lambek calculus with bracket modalities, is Morrill et al. (2018). It remains to investigate also bracket induction for the universal subexponential, and to incorporate full bracket induction into *CatLog*.

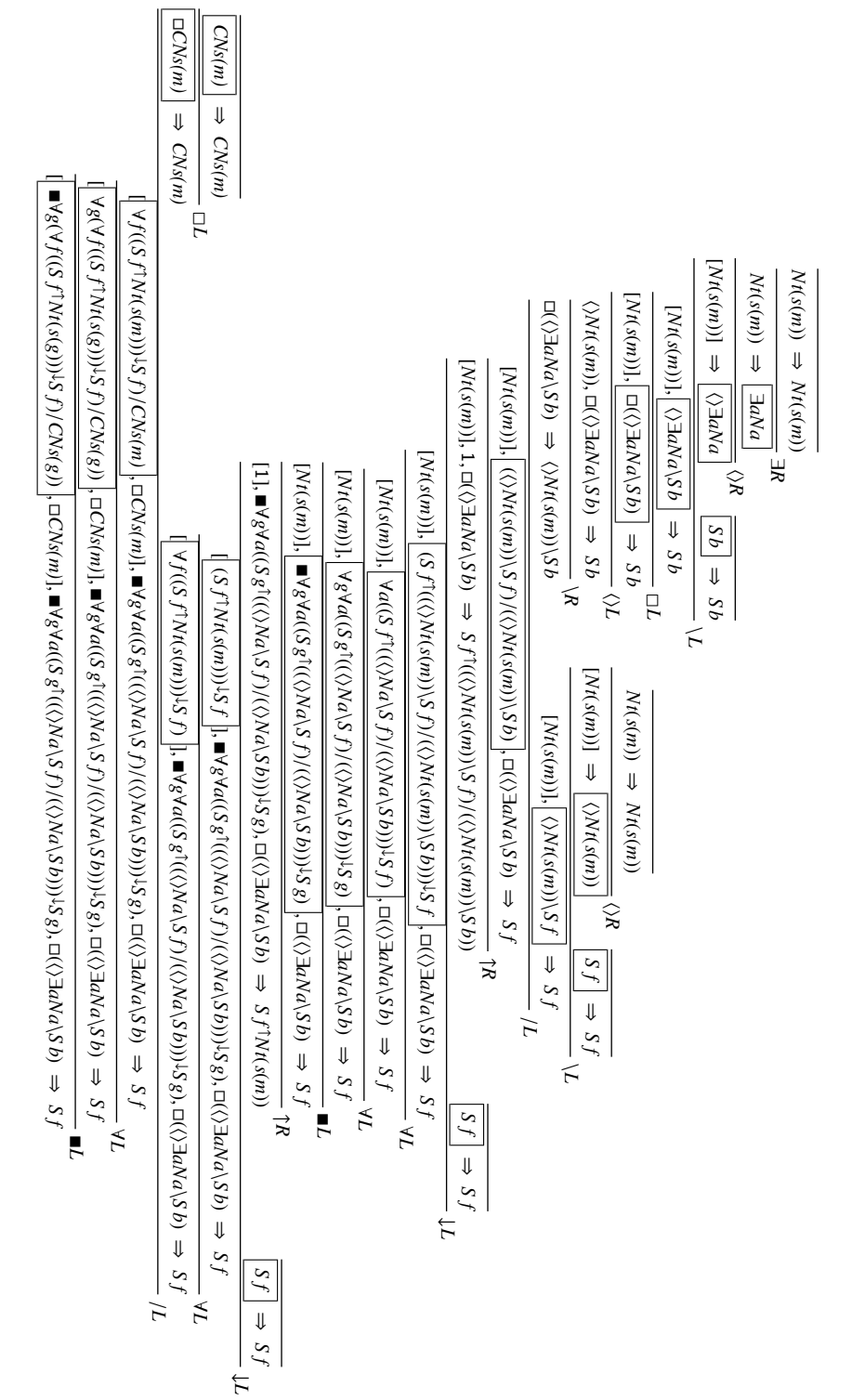

Fig. 20 Derivation of *Every man doesn't walk*, $\forall > \neg$ reading

Fig. 21 Derivation of *Every man doesn't walk*, ¬ > ∀ reading

Appendix: Relativisation examples

In this appendix we illustrate further by presenting the *CatLog3* LaTeX output for relativisation extraction and parasitic extraction examples. This comprises output which is unedited, except for the resizing of derivations into sideways figures; the details can be zoomed online.

(eac(rel(9))) **man**+[[**that**+[**mary**]+**likes**+**today**]] : $CNs(m)$

$\Box CNs(m) : man, [[\blacksquare\forall n([]^{-1}[]^{-1}(CNn\backslash CNn)/\blacksquare((\langle\rangle Nt(n)\sqcap !\blacksquare Nt(n))\backslash S\,f)) :$
$\lambda A\lambda B\lambda C[(B\ C)\wedge(A\ C)], [\blacksquare Nt(s(f)) : m], \Box((\langle\rangle\exists gNt(s(g))\backslash S\,f)/\exists aNa) :$
$\hat{}\lambda D\lambda E(Pres\ ((\check{}like\ D)\ E)), \Box\forall a\forall f(((\langle\rangle Na\backslash S\,f)\backslash(\langle\rangle Na\backslash S\,f)) : \hat{}\lambda F\lambda G(\check{}today\ (F\ G))]]$
$\Rightarrow\ CNs(m)$

See Figure 22.

$\lambda C[(\check{}man\ C)\wedge(\check{}today\ (Pres\ ((\check{}like\ C)\ m)))]$

(eac(rel(10))) **man**+**that**+[**the**+**friends**+**of**]+**walk** : $CNs(m)$

$\Box CNs(m) : man, \blacksquare\forall n([]^{-1}[]^{-1}(CNn\backslash CNn)/\blacksquare((\langle\rangle Nt(n)\sqcap !\blacksquare Nt(n))\backslash S\,f)) :$
$\lambda A\lambda B\lambda C[(B\ C)\wedge(A\ C)], [\blacksquare\forall n(Nt(n)/CNn) : \iota,$
$\Box(CNp/PPof) : friends, \Box((\forall n(CNn\backslash CNn)/\blacksquare\exists bNb)\&(PPof/\exists aNa)) :$
$\hat{}(\check{}of, \lambda DD)], \Box(\langle\rangle(\exists aNa{-}\exists gNt(s(g)))\backslash S\,f) : \hat{}\lambda E(Pres\ (\check{}walk\ E))\ \Rightarrow\ CNs(m)$

$\Box CNs(m) : man, \blacksquare\forall n([]^{-1}[]^{-1}(CNn\backslash CNn)/\blacksquare((\langle\rangle Nt(n)\sqcap !\blacksquare Nt(n))\backslash S\,f)) :$
$\lambda A\lambda B\lambda C[(B\ C)\wedge(A\ C)], [\blacksquare\forall n(Nt(n)/CNn) : \iota,$
$\Box(CNp/PPof) : friends, \Box((\forall n(CNn\backslash CNn)/\blacksquare\exists bNb)\&(PPof/\exists aNa)) :$
$\hat{}(\check{}of, \lambda DD)], \Box(\langle\rangle\exists aNa\backslash S\,b) : \hat{}\lambda E(\check{}walk\ E)\ \Rightarrow\ CNs(m)$

(eac(rel(11))) **man**+[[**that**+[[**the**+**friends**+**of**]]+**admire**]] : $CNs(m)$

$\Box CNs(m) : man, [[\blacksquare\forall n([]^{-1}[]^{-1}(CNn\backslash CNn)/\blacksquare((\langle\rangle Nt(n)\sqcap !\blacksquare Nt(n))\backslash S\,f)) :$
$\lambda A\lambda B\lambda C[(B\ C)\wedge(A\ C)], [[\blacksquare\forall n(Nt(n)/CNn) : \iota,$
$\Box(CNp/PPof) : friends, \Box((\forall n(CNn\backslash CNn)/\blacksquare\exists bNb)\&(PPof/\exists aNa)) : \hat{}(\check{}of, \lambda DD)]],$
$\Box((\langle\rangle(\exists aNa{-}\exists gNt(s(g)))\backslash S\,f)/\exists aNa) : \hat{}\lambda E\lambda F(Pres\ ((\check{}admire\ E)\ F))]]\ \Rightarrow\ CNs(m)$

See Figure 23.

$\lambda C[(\check{}man\ C)\wedge(Pres\ ((\check{}admire\ C)\ (\iota\ (\check{}friends\ C))))]$

(eac(rel(12))) **paper**+[[**that**+[**john**]+**filed**+[[**without**+**reading**]]]] : $CNs(n)$

$\Box CNs(n) : paper, [[\blacksquare\forall n([]^{-1}[]^{-1}(CNn\backslash CNn)/\blacksquare((\langle\rangle Nt(n)\sqcap !\blacksquare Nt(n))\backslash S\,f)) :$
$\lambda A\lambda B\lambda C[(B\ C)\wedge(A\ C)], [\blacksquare Nt(s(m)) : j], \Box((\langle\rangle\exists gNt(s(g))\backslash S\,f)/\exists aNa) :$
$\hat{}\lambda D\lambda E(Past\ ((\check{}file\ D)\ E)), [[\blacksquare\forall a\forall f([]^{-1}((\langle\rangle Na\backslash S\,f)\backslash(\langle\rangle Na\backslash S\,f))/(\langle\rangle Na\backslash S\,psp)) :$
$\lambda F\lambda G\lambda H[(G\ H)\wedge\neg(F\ H)], \Box((\langle\rangle\exists aNa\backslash S\,psp)/\exists aNa) : \hat{}\lambda I\lambda J((\check{}read\ I)\ J)]]]]$
$\Rightarrow\ CNs(n)$

See Figure 24.

Fig. 22 Derivation of *man that Mary likes today*

Fig. 23 Derivation of *man that the friends of admire*

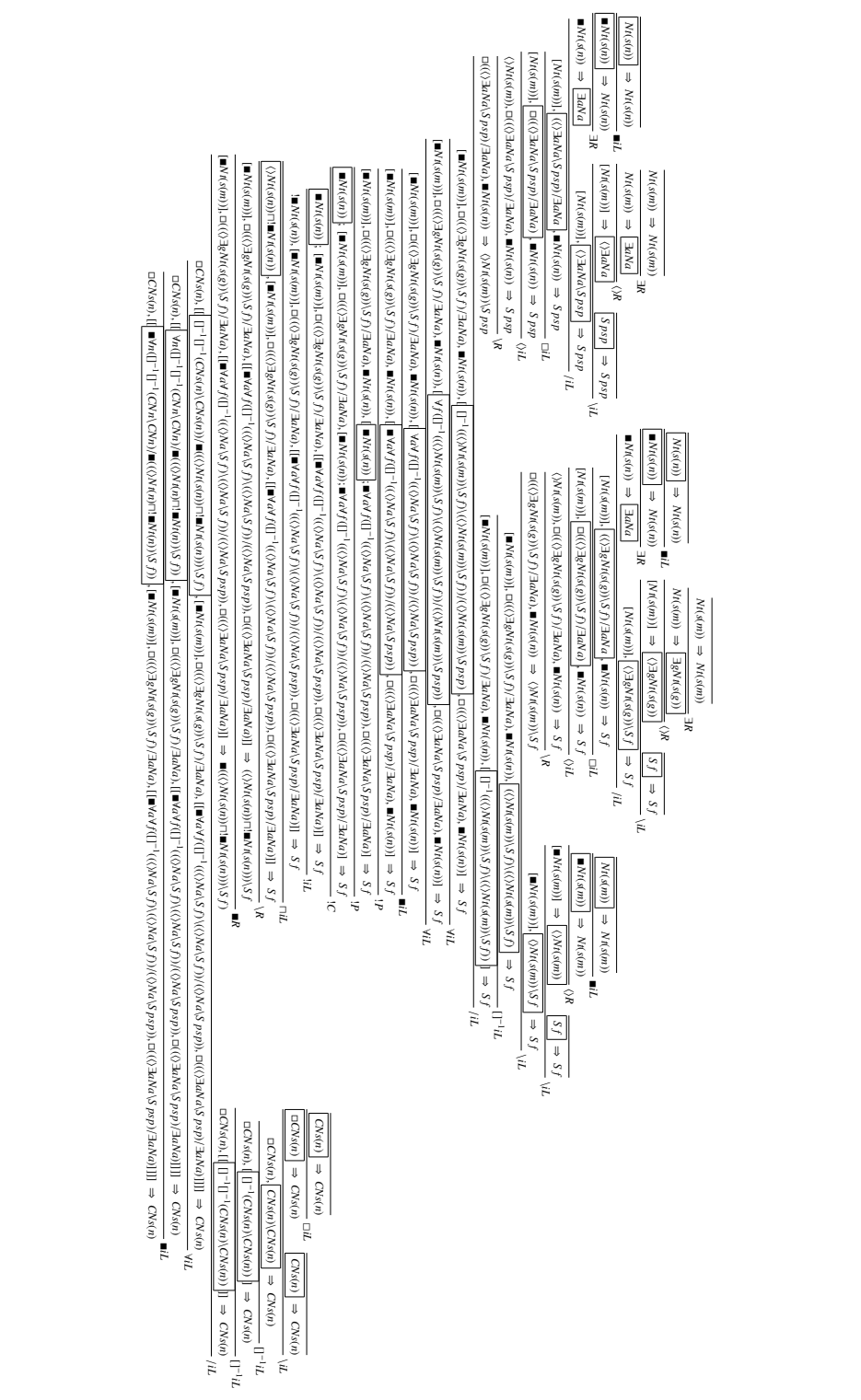

Fig. 24 Derivation of *paper that John filed without reading*

$\lambda C[(\check{}paper\ C) \wedge [(Past\ ((\check{}file\ C)\ j)) \wedge \neg((\check{}read\ C)\ j)]]$

(eac(rel(13))) **paper**+[[**that**+[[**the**+**editor**+**of**]]+**filed**+[[**without**+**reading**]]]] : $CNs(n)$

$\Box CNs(n) : paper, [[\blacksquare\forall n([]^{-1}[]^{-1}(CNn\backslash CNn)/\blacksquare((\langle\rangle Nt(n) \sqcap !\blacksquare Nt(n))\backslash S\ f)) :$
$\lambda A\lambda B\lambda C[(B\ C) \wedge (A\ C)], [[\blacksquare\forall n(Nt(n)/CNn) : \iota, \Box(\forall gCNs(g)/PPof) : editor,$
$\Box((\forall n(CNn\backslash CNn)/\blacksquare\exists bNb)\&(PPof/\exists aNa)) : \hat{}(\check{}of, \lambda DD)]],$
$\Box((\langle\rangle\exists gNt(s(g))\backslash S\ f)/\exists aNa) : \hat{}\lambda E\lambda F(Past\ ((\check{}file\ E)\ F)),$
$[[\blacksquare\forall a\forall f([]^{-1}((\langle\rangle Na\backslash S\ f)\backslash(\langle\rangle Na\backslash S\ f))/(\langle\rangle Na\backslash S\ psp)) : \lambda G\lambda H\lambda I[(H\ I) \wedge \neg(G\ I)],$
$\Box((\langle\rangle\exists aNa\backslash S\ psp)/\exists aNa) : \hat{}\lambda J\lambda K((\check{}read\ J)\ K)]]]] \Rightarrow CNs(n)$

See Figure 25.

$\lambda C[(\check{}paper\ C) \wedge [(Past\ ((\check{}file\ C)\ (\iota\ (\check{}editor\ C)))) \wedge \neg((\check{}read\ C)\ (\iota\ (\check{}editor\ C)))]]$

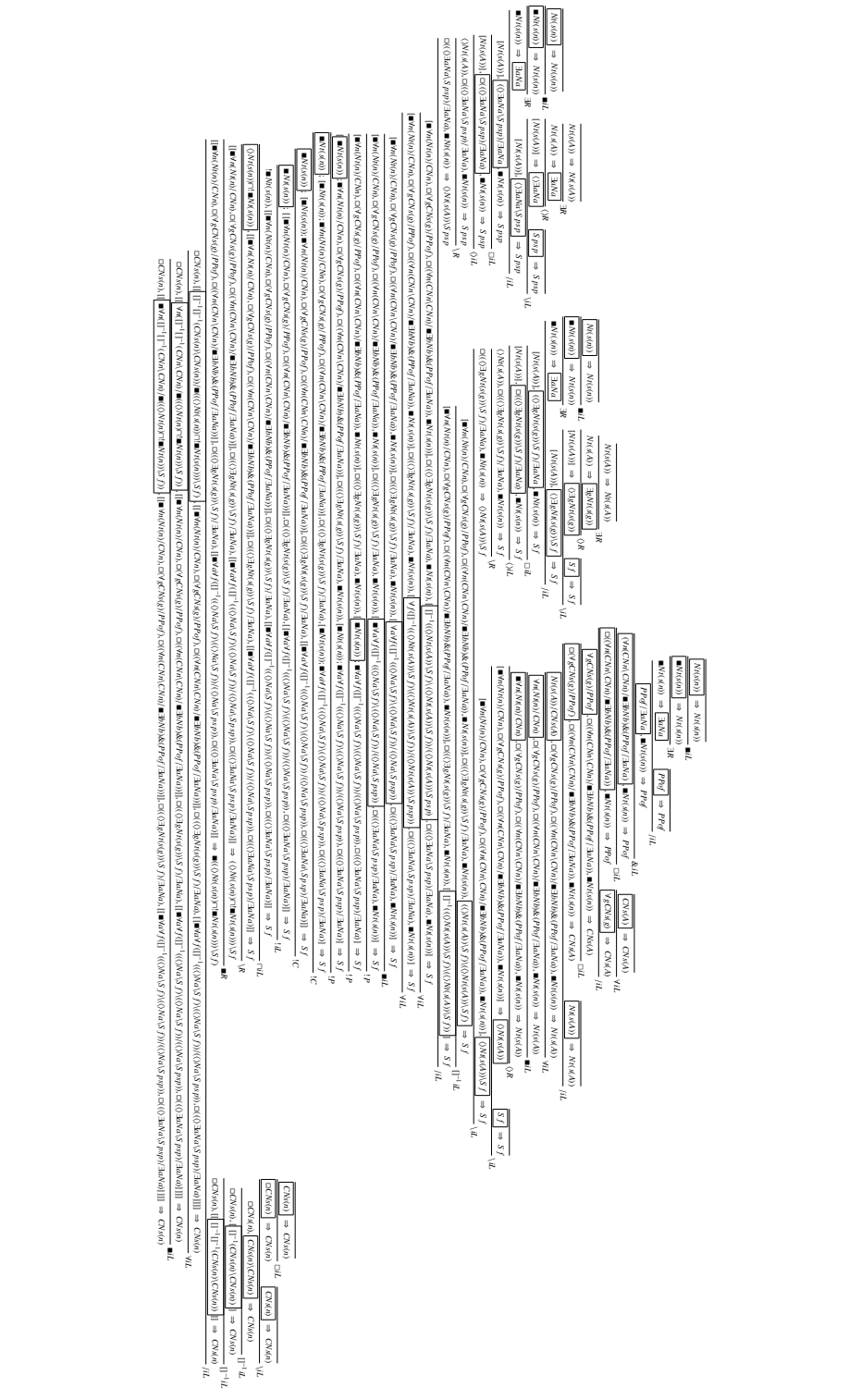

Fig. 25 Derivation of *paper that the editor of filed without reading*

References

Ajdukiewicz, K. (1935). Die syntaktische Konnexität. *Studia Philosophica 1*, 1–27. Translated in Storrs McCall, editor, 1967, *Polish Logic: 1920–1939*, Oxford University Press, Oxford, 207–231.

Andreoli, J.-M. (1992). Logic programming with focusing in linear logic. *Journal of Logic and Computation 2*(3), 297–347.

Bar-Hillel, Y. (1953). A quasi-arithmetical notation for syntactic description. *Language 29*, 47–58.

Buszkowski, W. (2007). On action logic: equational theories of action algebras. *Journal of Logic and Computation 17*(1), 199–217.

Buszkowski, W. and E. Palka (2008). Infinitary action logic: complexity, models and grammars. *Studia Logica 89*(1), 1–18.

Carpenter, B. (1997). *Type-Logical Semantics*. Cambridge, MA: MIT Press.

Chaudhuri, K., D. Miller, and A. Saurin (2008). Canonical sequent proofs via multi-focusing. In G. Ausiello, J. Karhumäki, G. Mauri, and L. Ong (Eds.), *Fifth Ifip International Conference On Theoretical Computer Science – Tcs 2008*, Boston, MA, pp. 383–396. Springer US.

Chomsky, N. (1957). *Syntactic Structures*. The Hague: Mouton.

Dowty, D. R., R. E. Wall, and S. Peters (1981). *Introduction to Montague Semantics*, Volume 11 of *Synthese Language Library*. Dordrecht: D. Reidel.

Fadda, M. (2010). *Geometry of Grammar: Exercises in Lambek Style*. Ph. D. thesis, Universitat Politècnica de Catalunya, Barcelona.

Gentzen, G. (1934). Untersuchungen über das logische Schliessen. *Mathematische Zeitschrift 39*, 176–210 and 405–431. Translated in M.E. Szabo, editor, 1969, *The Collected Papers of Gerhard Gentzen*, North-Holland, Amsterdam, 68–131.

Girard, J.-Y. (1987). Linear logic. *Theoretical Computer Science 50*, 1–102.

Girard, J.-Y. (2011). *The Blind Spot*. Zürich: European Mathematical Society.

Hendriks, H. (1993). *Studied flexibility. Categories and types in syntax and semantics*. Ph. D. thesis, Universiteit van Amsterdam, ILLC, Amsterdam.

Hepple, M. (1990a). Normal form theorem proving for the Lambek calculus. In H. Karlgren (Ed.), *Proceedings of COLING*, Stockholm.

Hepple, M. (1990b). *The Grammar and Processing of Order and Dependency*. Ph. D. thesis, University of Edinburgh.

Jäger, G. (2005). *Anaphora and Type Logical Grammar*, Volume 24 of *Trends in Logic – Studia Logica Library*. Dordrecht: Springer.

Kanazawa, M. (1992). The Lambek calculus enriched with additional connectives. *Journal of Logic, Language and Information 1*, 141–171.

Kanovich, M., S. Kuznetsov, and A. Scedrov (2017). Undecidability of the lambek calculus with subexponential and bracket modalities. In *Proc. FCT*, Volume 10472 of *LNCS*, pp. 326–340.

König, E. (1989). Parsing as natural deduction. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, Vancouver.

Kuznetsov, S., G. Morrill, and O. Valentín (2017). Count-invariance including exponentials. In M. Kanazawa (Ed.), *Mathematics of Language*, London.

Lafont, Y. (2004, 6 June). Soft linear logic and polynomial time. *Theoretical Computer Science 318*(12), 163–180.

Lambek, J. (1958). The mathematics of sentence structure. *American Mathematical Monthly 65*, 154–170.

Lambek, J. (1961). On the Calculus of Syntactic Types. In R. Jakobson (Ed.), *Structure of Language and its Mathematical Aspects, Proceedings of the Symposia in Applied Mathematics XII*, pp. 166–178. Providence, Rhode Island: American Mathematical Society.

Lambek, J. (1988). Categorial and Categorical Grammars. In R. T. Oehrle, E. Bach, and D. Wheeler (Eds.), *Categorial Grammars and Natural Language Structures*, Volume 32 of *Studies in Linguistics and Philosophy*, pp. 297–317. Dordrecht: D. Reidel.

Miller, D., G. Nadathur, F. Pfenning, and A. Scedrov (1991). Uniform proofs as a foundation for logic programming. *Annals of Pure and Applied Logic 51*(1-2), 125–157.

Montague, R. (1973). The Proper Treatment of Quantification in Ordinary English. In J. Hintikka, J. Moravcsik, and P. Suppes (Eds.), *Approaches to Natural Language: Proceedings of the 1970 Stanford Workshop on Grammar and Semantics*, pp. 189–224. Dordrecht: D. Reidel. Reprinted in R.H. Thomason, editor, 1974, *Formal Philosophy: Selected Papers of Richard Montague*, Yale University Press, New Haven, 247–270.

Moortgat, M. (1988). *Categorial Investigations: Logical and Linguistic Aspects of the Lambek Calculus*. Foris, Dordrecht. PhD thesis, Universiteit van Amsterdam.

Moortgat, M. (1996). Multimodal linguistic inference. *Journal of Logic, Language and Information 5*(3, 4), 349–385. Also in *Bulletin of the IGPL*, 3(2,3):371–401, 1995.

Moortgat, M. (1997). Categorial Type Logics. In J. van Benthem and A. ter Meulen (Eds.), *Handbook of Logic and Language*, pp. 93–177. Amsterdam and Cambridge, Massachusetts: Elsevier Science B.V. and the MIT Press.

Moortgat, M. and R. Moot (2013). Proof nets for the Lambek-Grishin calculus. In E. Grefenstette, C. Heunen, and M. Sadrzadeh (Eds.), *Compositional Methods in Physics and Linguistics*. Oxford University Press. To appear.

Moot, R. (2014). Extended Lambek Calculi and First-Order Linear Logic. In M. M. Claudia Casadio, Bob Coeke and P. Scott (Eds.), *Categories and Types in Logic, Language and Physics: Essays Dedicated to Jim Lambek on the Occasion of His 90th Birthday*, Volume 8222 of *LNCS, FoLLI Publications in Logic, Language and Information*, pp. 297–330. Berlin: Springer.

Moot, R. (2016). Proof nets for the displacement calculus. In A. Foret, G. , R. Muskens, R. Osswald, and S. Pogodalla (Eds.), *Formal Grammar: 20th and 21st International Conferences*, Volume 9804 of *LNCS, FoLLI Publications in Logic, Language and Information*, Berlin, pp. 273–289. Springer.

Moot, R. and C. Retoré (2012). *The Logic of Categorial Grammars: A Deductive Account of Natural Language Syntax and Semantics*. Heidelberg: Springer.

Morrill, G. (1990). Intensionality and Boundedness. *Linguistics and Philosophy 13*(6), 699–726.

Morrill, G. (1992). Categorial Formalisation of Relativisation: Pied Piping, Islands, and Extraction Sites. Technical Report LSI-92-23-R, Departament de Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya.

Morrill, G. (2011a). Logic Programming of the Displacement Calculus. In S. Pogodalla and J.-P. Prost (Eds.), *Proceedings of Logical Aspects of Computational Linguistics 2011, LACL'11, Montpellier*, Number LNAI 6736 in Springer Lecture Notes in AI, Berlin, pp. 175–189. Springer.

Morrill, G. (2012). CatLog: A Categorial Parser/Theorem-Prover. In *LACL 2012 System Demonstrations*, Logical Aspects of Computational Linguistics 2012, Nantes, pp. 13–16.

Morrill, G. (2017). Grammar logicised: relativisation. *Linguistics and Philosophy 40*(2), 119–163.

Morrill, G. and M. Fadda (2008). Proof Nets for Basic Discontinuous Lambek Calculus. *Logic and Computation 18*(2), 239–256.

Morrill, G., S. Kuznetsov, M. Kanovich, and A. Scedrov (2018). Bracket induction for lambek calculus with bracket modalities. In Foret, Kobele, and Pogodalla (Eds.), *Proceedings of Formal Grammar 2018, Sofia*, Berlin, pp. 84–101. Springer-Verlag.

Morrill, G. and J.-M. Merenciano (1996). Generalising Discontinuity. *Traitement automatique des langues 37*(2), 119–143.

Morrill, G. and O. Valentín (2010). Displacement Calculus. *Linguistic Analysis 36*(1–4), 167–192. Special issue Festschrift for Joachim Lambek.

Morrill, G. and O. Valentín (2014). Semantically Inactive Multiplicatives and Words as Types. In N. Asher and S. Soloviev (Eds.), *Proceedings of Logical Aspects of Computational Linguistics, LACL'14, Toulouse*, Number 8535 in LNCS, FoLLI Publications on Logic, Language and Information, Berlin, pp. 149–162. Springer.

Morrill, G. and O. Valentín (2016). Computational coverage of Type Logical Grammar: The Montague Test. In C. Piñón (Ed.), *Empirical Issues in Syntax and Semantics*, Volume 11, pp. 141–170. Paris: Colloque de Syntaxe et Sémantique à Paris (CSSP).

Morrill, G. and O. Valentín (2018). Spurious ambiguity and focalization. *Computational Linguistics 44*(2), 285–327.

Morrill, G., O. Valentín, and M. Fadda (2009). Dutch Grammar and Processing: A Case Study in TLG. In P. Bosch, D. Gabelaia, and J. Lang (Eds.), *Logic, Language, and Computation: 7th International Tbilisi Symposium, Revised Selected Papers*, Number 5422 in Lecture Notes in Artificial Intelligence, Berlin, pp. 272–286. Springer.

Morrill, G., O. Valentín, and M. Fadda (2011). The Displacement Calculus. *Journal of Logic, Language and Information 20*(1), 1–48.

Morrill, G. V. (1994). *Type Logical Grammar: Categorial Logic of Signs*. Dordrecht: Kluwer Academic Publishers.

Morrill, G. V. (2011b). *Categorial Grammar: Logical Syntax, Semantics, and Processing*. New York and Oxford: Oxford University Press.

Valentín, O. (2012). *Theory of Discontinuous Lambek Calculus*. Ph. D. thesis, Universitat Autònoma de Barcelona, Barcelona.

Valentín, O., D. Serret, and G. Morrill (2013). A Count Invariant for Lambek Calculus with Additives and Bracket Modalities. In G. Morrill and M.-J. Nederhof (Eds.), *Proceedings of Formal Grammar 2012 and 2013*, Volume 8036 of *Springer LNCS, FoLLI Publications in Logic, Language and Information*, Berlin, pp. 263–276. Springer.

van Benthem, J. (1991). *Language in Action: Categories, Lambdas, and Dynamic Logic*. Number 130 in Studies in Logic and the Foundations of Mathematics. Amsterdam: North-Holland. Revised student edition printed in 1995 by the MIT Press.