NL_λ as the logic of scope and movement

Chris Barker, *New York University*, cb125@nyu.edu

April 26, 2018

Abstract

FIRST DRAFT plus minor revisions. Lambek elegantly characterized the structure of natural language, but he only finished half the job. It's well-known that his substructural logic L, and its non-associative version NL, handle basic composition well ('external merge'), but not scope taking and syntactic displacement ('internal merge')—at least, not in their full generality. In previous work, I propose NL_λ, which is NL supplemented with a single structural inference rule ("abstraction"). Abstraction closely resembles the traditional linguistic rule of Quantifier Raising, and characterizes both semantic scope taking and syntactic displacement. Due to the unconventional shape of the abstraction inference, there has been some doubt that NL_λ is well-behaved as a logic. This paper puts such worries to rest. I prove that the logic enjoys cut elimination as well as interpolation (roughly: every structure can be cut). In addition, perhaps surprisingly, I prove NL_λ is decidable. Finally, I prove that it is sound and complete with respect to the usual class of relational frames.

Keywords: Lambek, substructural logic, scope, syntactic movement, continuations, decidability

Thanks to Glyn Morrill and Greg Restall for helpful discussion.

1 NL_λ

NL_λ is a substructural logic (Restall 2000, Ono 2003) with applications in the syntax and semantics of natural languages. It is based on the non-associative Lambek calculus NL (Lambek 1958, 1961; Moortgat 1997, Buszkowski 2010), augmented by a structural postulate ("abstraction") that corresponds to the traditional linguistic rule of Quantifier Raising (May 1977, 1985; Heim and Kratzer 1998).

NL_λ was first proposed in Barker 2007a, where it was used to provide a compositional account of the semantics of *same* and *different* in English. The present paper concentrates on the formal properties of NL_λ, rather than on its linguistic applications; for analyses in NL_λ of scope, binding, ellipsis, and more, see chapters 13 through 16 in Barker and Shan 2014, as well as Barker 2007a, 2013, 2015a, 2015b, and Barker in prep.

Because the abstraction postulate does not resemble standard structural postulates (for a list of standard structural postulates, see, e.g., Restall 2000:26), there has been some doubt that NL_λ is a legitimate substructural logic. For instance, Kubota 2015:23 remarks that "[t]he use of λs in

structured antecedents is without precedent," and he questions the "formal and ontological status" of the postulate. So the present paper sets out to establish that although NL_λ is indeed innovative, it is nevertheless perfectly legitimate and well-behaved as a substructural logic.

The mistrust of the abstraction postulate persists despite the fact that Barker and Shan 2014 (section 17.9) prove soundness and completeness and decidability for a restricted version of NL_λ. However, there are two aspects of that discussion that limit its persuasiveness, both of which are remedied here. One limitation is that their proof is indirect: they first prove that NL_λ is equivalent to a more conventional substructural logic, and then prove soundness and completeness for the second logic. The proofs here deal directly with NL_λ. The second limitation is that because their results apply only to a restricted version of NL_λ, they leave open the status of unrestricted NL_λ. They remark (page 181) that extending the results to the unrestricted version would require "new techniques". The present paper supplies the needed new techniques.

The logic has two modes. One mode corresponds to Lambek's original logic, and characterizes horizontal composition, i.e., left/right adjacency (in syntactic terms, "external merge"). The other mode characterizes vertical composition, that is, context/value combination ("internal merge"). The vertical mode allows an expression to combine with one of its delimited continuations, so I will sometimes call this the continuation mode.

The addition of a product unit (as in, e.g., Lambek 1988) for the horizontal mode allows the logic to model gaps left by syntactic movement. Thus abstraction and the unit together allow NL_λ to characterize both covert movement (scope) and overt movement (syntactic displacement).

Unrestricted NL_λ, the logic of scope and movement:

- Formulas: $F := \mathrm{DP} \mid \mathrm{S} \mid \mathrm{Q} \mid t \mid F\backslash F \mid F \times F \mid F/F \mid F \mathbin{\backslash\!\!\backslash} F \mid F \otimes F \mid F \mathbin{/\!\!/} F$
- Basic structures (see text for explanation): $S := F \mid 1 \mid S \bullet S \mid S \circ S$
- Axiom and cut:

$$\frac{}{A \vdash A}\ \mathrm{AXIOM} \qquad \frac{\Gamma \vdash A \quad \Sigma[A] \vdash B}{\Sigma[\Gamma] \vdash B}\ \mathrm{CUT}$$

- Logical inference rules:

$$\frac{\Gamma \vdash A \quad \Sigma[B] \vdash C}{\Sigma[\Gamma \bullet A\backslash B] \vdash C}\backslash L \qquad \frac{A \bullet \Gamma \vdash B}{\Gamma \vdash A\backslash B}\backslash R \qquad \frac{\Gamma \vdash A \quad \Sigma[B] \vdash C}{\Sigma[\Gamma \circ A \mathbin{\backslash\!\!\backslash} B] \vdash C} \mathbin{\backslash\!\!\backslash} L \qquad \frac{A \circ \Gamma \vdash B}{\Gamma \vdash A \mathbin{\backslash\!\!\backslash} B} \mathbin{\backslash\!\!\backslash} R$$

$$\frac{\Sigma[A \bullet B] \vdash C}{\Sigma[A \times B] \vdash C} \times L \qquad \frac{\Gamma \vdash A \quad \Sigma \vdash B}{\Gamma \bullet \Sigma \vdash A \times B} \times R \qquad \frac{\Sigma[A \circ B] \vdash C}{\Sigma[A \otimes B] \vdash C} \otimes L \qquad \frac{\Gamma \vdash A \quad \Sigma \vdash B}{\Gamma \circ \Sigma \vdash A \otimes B} \otimes R$$

$$\frac{\Gamma \vdash A \quad \Sigma[B] \vdash C}{\Sigma[B/A \bullet \Gamma] \vdash C} /L \qquad \frac{\Gamma \bullet A \vdash B}{\Gamma \vdash B/A} /R \qquad \frac{\Gamma \vdash A \quad \Sigma[B] \vdash C}{\Sigma[B \mathbin{/\!\!/} A \circ \Gamma] \vdash C} \mathbin{/\!\!/} L \qquad \frac{\Gamma \circ A \vdash B}{\Gamma \vdash B \mathbin{/\!\!/} A} \mathbin{/\!\!/} R$$

$$\frac{\Sigma[1] \vdash A}{\Sigma[t] \vdash A}\ \mathrm{t}L \qquad \frac{}{1 \vdash t}\ \mathrm{t}R$$

- Structural rules:

$$1 \bullet \Delta \equiv \Delta \equiv \Delta \bullet 1 \qquad \text{(unit)}$$
$$\Sigma[\Delta] \equiv \Delta \circ \lambda\alpha\Sigma[\alpha] \qquad \text{(abstraction)}$$

In view of the structural rule governing lambda structures ("abstraction"), we must enlarge the set of basic structures by adding the following formation rule: if Δ and $\Sigma[\Delta]$ are structures, and α is a variable from the set $\{x, y, z, \ldots\}$ not occurring in $\Sigma[\,]$, then $\lambda\alpha\Sigma[\alpha]$ is a structure. For instance, DP, DP $\bullet$ S, DP $\circ\, \lambda x(x \bullet \text{S})$, and $\lambda y \lambda x(x \bullet y)$ are structures, but x and $(x \bullet \text{S})$ are not. Note that by construction, these lambda structures are linear in the following sense: for every $\lambda\alpha$, there is exactly one occurrence of α in the body of the structure.

The abstraction inference rule closely resembles the traditional linguistic rule of Quantifier Raising (May 1977, 1985; Heim and Kratzer 1998). It says that whenever a structure Σ contains within it an occurrence of the structure Δ, Δ can potentially take scope over Σ. This is accomplished here by adjoining Δ in the continuation mode (i.e., via '$\circ$') with the lambda structure $\lambda\alpha\Sigma[\alpha]$, where α is a variable not occurring anywhere in $\Sigma[\,]$. Since the lambda structure $\lambda\alpha\Sigma[\alpha]$ surrounds Δ, it constitutes one of Δ's (delimited) continuations.

As we should expect from the resemblance with Quantifier Raising, NL$_\lambda$ is well suited to modeling in-situ quantifier scope. If we assign *Ann* to the category DP, *saw* to the category $(\text{DP}\backslash\text{S})/\text{DP}$, and the quantifier *everyone* to the category $\text{S}/\!\!/(\text{DP}\backslash\!\!\backslash\text{S})$, we have the following account of in-situ scope taking:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\text{DP} \vdash \text{DP} \quad \text{S} \vdash \text{S}}{\text{DP} \bullet \text{DP}\backslash\text{S} \vdash \text{S}}\backslash L \quad \text{DP} \vdash \text{DP}}{\text{DP} \bullet ((\text{DP}\backslash\text{S})/\text{DP} \bullet \text{DP}) \vdash \text{S}}/L}{\text{DP} \circ \lambda x(\text{DP} \bullet ((\text{DP}\backslash\text{S})/\text{DP} \bullet x)) \vdash \text{S}}\text{ABS}}{\lambda x(\text{DP} \bullet ((\text{DP}\backslash\text{S})/\text{DP} \bullet x)) \vdash \text{DP}\backslash\!\!\backslash\text{S}}\backslash\!\!\backslash R \quad \text{S} \vdash \text{S}}{\text{S}/\!\!/(\text{DP}\backslash\!\!\backslash\text{S}) \circ \lambda x(\text{DP} \bullet ((\text{DP}\backslash\text{S})/\text{DP} \bullet x)) \vdash \text{S}}/\!\!/L}{\text{DP} \bullet ((\text{DP}\backslash\text{S})/\text{DP} \bullet \text{S}/\!\!/(\text{DP}\backslash\!\!\backslash\text{S})) \vdash \text{S}}\text{ABS}}{\text{Ann} \bullet (\text{saw} \bullet \text{everyone}) \vdash \text{S}}\text{LEX}$$

The natural Curry-Howard labeling (see, e.g., chapter 13 in Barker and Shan 2014) gives the desired semantics for scope-taking, namely, $\mathbf{everyone}(\lambda x.\mathbf{saw}(x)(\mathbf{ann}))$.

One way to understand the abstraction rule is as a kind of syntactic movement. In this derivation, the scope-taker *everyone* moves upwards to adjoin to its scope domain. A movement analogy is not necessary, however. It is equally legitimate to say that the scope-taker remains in place, and we are simply focussing it, and placing the context that surrounds it into the background. See Barker and Shan 2014, Barker 2015a, 2015b for discussion.

With the addition of a unit for the horizontal mode, the logic also provides an account of overt syntactic movement. The unit structural rule says that the structural constant '1' (which can be thought of as the empty structure, '()') serves as both a left and right unit for the horizontal mode. The unit will be used to model gaps for overt movement as well as silent modifiers, and plays

a key role in sprouting in the analysis of sluicing in Barker 2013 and chapter 15 of Barker and Shan 2014. The unit structure corresponds to the type 't'. The symbol 't' stands for the weakest tautology, as in, e.g., Restall 2000, but in a happy coincidence, it also corresponds to the symbol often used in the syntactic literature to stand for syntactic traces.

For example, if we assign *who* to the category $\mathrm{Q}/(t\otimes(\mathrm{DP}\backslash\!\backslash\mathrm{S}))$, then following Morrill et al. 2011:22, we can analyze an embedded question such as *who Ann saw*, as in *Bill knows [who Ann saw]* as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{1\vdash t}\qquad \lambda x(\mathrm{Ann}\bullet(\mathrm{saw}\bullet x))\vdash \mathrm{DP}\backslash\!\backslash\mathrm{S}}{1\circ\lambda x(\mathrm{Ann}\bullet(\mathrm{saw}\bullet x))\vdash t\otimes(\mathrm{DP}\backslash\!\backslash\mathrm{S})}\otimes R}{\mathrm{Ann}\bullet(\mathrm{saw}\bullet 1)\vdash t\otimes(\mathrm{DP}\backslash\!\backslash\mathrm{S})}\,\mathrm{ABS}}{\mathrm{Ann}\bullet\mathrm{saw}\vdash t\otimes(\mathrm{DP}\backslash\!\backslash\mathrm{S})}\,\mathrm{unit}\qquad \mathrm{Q}\vdash\mathrm{Q}}{\mathrm{Q}/(t\otimes(\mathrm{DP}\backslash\!\backslash\mathrm{S}))\bullet(\mathrm{Ann}\bullet\mathrm{saw})\vdash\mathrm{Q}}\backslash L$$

In the traditional syntactic treatment, the wh-word *who* moves from the object position of *saw*, leaving a trace. That movement corresponds here to the abstraction of the unit, establishing a logical connection between the object position and the wh-word. The t in the lexical category of *who* licenses the abstraction of the unit.

As mentioned above, Barker and Shan 2014 place restrictions on the abstraction rule that are not imposed here (see their section 17.6, page 190). Essentially, they allow abstraction only out of contexts built from the horizontal mode, but not out of contexts built from the vertical mode. This restriction allows Barker and Shan to prove an Efficient Abstraction theorem, which says that for every derivation, there is an equivalent derivation in which only formulas (as opposed to complex structures) undergo abstraction (i.e., Δ in the abstraction rule is always a formula). As they note, this has the unfortunate effect of ruling out derivations proposed for sluicing in Barker 2013 (see discussion in Barker and Shan 2014:205, section 7.11), since sluicing crucially requires abstraction of a lambda structure. The unrestricted version of NL$_\lambda$ given here covers all of the sluicing analyses proposed in Barker 2013, and the decidability proof does not depend on Efficient Abstraction.

The presence of lambdas and variables in the syntax of NL$_\lambda$ often creates concern; for an example in print, once again, see Kubota 2015:23. Are these lambdas "real" lambdas? And what about the variables? Although these formal elements are novel in the context of the syntactic side of Lambek grammars, variables and binding constructions are common in logic in general (just think of the Predicate Calculus with quantification). Furthermore, there is a related literature that uses lambda expressions in syntactic calculi. Notable examples include Oehrle 1994, Muskens 2001, and de Groote 2002. So my strong inclination is to reply, "Sure, they're real lambdas, why not?" Certainly, the lambda abstracts here always behave exactly as one would expect a real lambda to behave. However, my official answer is "It doesn't matter": the set of formulas in the logic are well-defined, the set of structures in the logic are well-defined, and the inferences licensed by the abstraction postulate are well-defined. Just like any structural rule, the abstraction rule licenses replacing one structure with a different structure based on purely structural criteria.

In other words, just like any logic, NL_λ fully and precisely characterizes a class of inferences. And just as we can for any logic, we can ask the usual metalogical questions: does the logic enjoy cut elimination? (Yes.) Is the logic decidable? (Surprisingly, yes.) Is there an interpolation theorem? (Yes.) What sort of semantics fits the logic? (The same relational frames often used to provide a semantics for other substructural logics.) Is the logic sound and complete with respect to its semantics? (Yes.)

2 Decidability

Theorem 1 ("decidability"): NL_λ is decidable.

This is not self-evident, since abstraction inferences can be repeated ad infinitum (e.g., $A \equiv A \circ \lambda xx \equiv (A \circ \lambda xx) \circ \lambda xx \equiv ...$). Without some way of limiting abstraction, proof search will continue forever. Proving decidability will involve three main lemmas: cut elimination, a bound on the number of abstractions, and a bound on the number of units.

2.1 Cut elimination lemma

NL_λ enjoys cut elimination.

Lemma (cut elimination): for any derivation starting from axioms that contains one or more instances of the cut inference, there is an equivalent proof (same final sequent, same Curry-Howard labeling) that does not use cut.

I will not prove cut elimination here, since there is a detailed proof in Barker and Shan 2014 chapter 17 that applies to unrestricted NL_λ. However, it is easy to see why cut elimination goes through for NL_λ. The standard proof of cut admissibility relies on pushing cuts upward in the proof until one of the premises is an axiom, at which point both the axiom and the cut can be eliminated without disturbing the proof. The logical rules of NL_λ are the same as the rules for other Lambek-style grammars, and so are equally compatible with cut elimination. As for the abstraction postulate, since every formula present in the conclusion of an abstraction is also present in its premise, any cut that targets a formula in the conclusion can be pushed upwards to target the corresponding formula in the premise, so abstraction inferences do not impede cut elimination.

I will henceforth assume that if a sequent is provable in NL_λ, then there is a cut-free proof.

2.2 Abstraction lemma

Since the abstraction inference rule is bi-directional, we can separate it into two parts, corresponding to beta reduction ('RED') and beta expansion ('EXP'):

$$\frac{\Sigma[\Gamma[\Delta]] \vdash A}{\Sigma[\Delta \circ \lambda x \Gamma[x]] \vdash A}\ \text{RED} \qquad \frac{\Sigma[\Delta \circ \lambda x \Gamma[x]] \vdash A}{\Sigma[\Gamma[\Delta]] \vdash A}\ \text{EXP}$$

The terms 'reduction' and 'expansion' reflect the bottom-to-top direction of fit, since decidability is a matter of proof search, which involves starting with the final conclusion and searching to find a proof that licenses it. The talk of beta reduction and so on is on analogy with the lambda calculus.

Since the premise of a reduction inference is strictly simpler than its conclusion, it poses no threat to decidability. The goal, then, will be to establish a bound on the number of expansions needed.

The strategy for proving the lemma stated immediately below will depend on tracking the lifetime within the proof of each '$\circ$'. Since there are no '$\circ$'s in axiom instances, each '$\circ$' must be introduced into the derivation either by a logical rule (more specifically, by $\backslash\!\backslash L$, $\otimes R$, or $/\!/ L$) or by a reduction inference. Once a '$\circ$' has been introduced, it will be copied by subsequent inference rules from premise to conclusion until it is eliminated either by a logical rule ($\backslash\!\backslash R$, $\otimes L$, or $/\!/ R$), or by an expansion inference. I'll say that a chain of $\circ$'s that is introduced by a reduction and eliminated by an expansion is a *purely structural* abstraction.

Lemma (purely structural abstraction is eliminable): given an arbitrary derivation in NL_λ, there is an equivalent derivation (same conclusion, same Curry-Howard labeling) in which no '$\circ$' eliminated by an expansion inference was introduced by a reduction inference.

Given the lemma, it follows that we can limit proof search to derivations in which each '$\circ$' eliminated by an expansion inference was introduced by a logical rule. Because each logical rule introduces at most one '$\circ$', and because the number of instances of the logical rules is bounded by the number of logical connectives in the final sequent, it follows that the number of expansion inferences is also bounded by the number of logical connectives in the final sequent.

Proof of the abstraction lemma: Since NL_λ enjoys cut elimination, we can assume without any loss of generality that we have a cut-free proof. If the derivation does not contain any purely structural abstractions, the derivation is already consistent with the lemma. Therefore assume that there is at least one purely structural abstraction in the proof. We will construct an equivalent parallel proof in which the purely structural abstraction is omitted.

The key fact is that the presence of the purely structural abstraction in the original derviation *limits* the range of available inferences rather than enlarges it. That is, any inference that can be performed on the original proof can be exactly replicated in the proof constructed without the abstraction.

To see why, consider the following schematic picture:

$$\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\Sigma[\Gamma[\Delta]] \vdash A}}{\Sigma[\Delta \circ \lambda x \Gamma[x]] \vdash A}\,\mathrm{RED}}{\vdots}}{\dfrac{\Sigma'[\Delta' \circ \Pi'[\lambda x \Gamma'[x]]] \vdash A'}{\Sigma''[\Delta'' \circ \Pi''[\lambda x \Gamma''[x]]] \vdash A''}\,\mathrm{X}}{\vdots}}{\dfrac{\Sigma'''[\Delta''' \circ \lambda x \Gamma'''[x]] \vdash A'''}{\Sigma'''[\Gamma'''[\Delta''']] \vdash A'''}\,\mathrm{EXP}}{\vdots}$$

$$\qquad\qquad \dfrac{\dfrac{\dfrac{\vdots}{\Sigma[\Gamma[\Delta]] \vdash A}}{\vdots}}{\dfrac{\Sigma'[\Pi'[\Gamma'[\Delta']]] \vdash A'}{\Sigma''[\Pi''[\Gamma''[\Delta'']]] \vdash A''}\,\mathrm{X}}{\vdots}}{\Sigma'''[\Gamma'''[\Delta''']] \vdash A'''}{\vdots}$$

In the original proof on the left, a reduction introduces a '$\circ$' that is carried through until a matching expansion eliminates it. The claim is that is always possible to construct a corresponding modified

proof as schematized on the right with identical starting and ending sequents, but without the abstraction.

Examining the reduction inference in the original derivation, note that in the conclusion of the inference, the '$\circ$' is adjacent to a corresponding λ. Subsequent inferences can interpose structure between a '$\circ$' and its lambda. For instance, additional reductions can abstract material from Γ into this position, and we can insert as many units as we wish. The possibility of interpolated material is schematically represented in the middle inference as Π' and Π''. However, in order for the final expansion to eliminate the '$\circ$', the '$\circ$' must be reunited with its original λ, since inspection of the logic reveals there is no way to create or maneuver any different lambda into the position adjacent to a pre-existing '$\circ$'.

Focus now on the schematic inference labelled X. The intention is that X can be any inference, either logical or structural (and so may have a second premise, not represented in the schematic diagram). Assuming that the premise in the original proof and the premise in the constructed proof share Σ', Π', Γ', and Δ', the premises differ only in the position of Δ', the presence of the $\circ$ and the λx, and the occurrence of x embedded in Γ'. In particular, every formula, every structural connective, and every potential landing site for an abstraction that is present in the left side premise is present also in the right side premise. It follows that the right hand premise will match the requirements of the X inference, and that the left hand conclusion and the right hand conclusion will also share identical Σ'', Π'', Γ'', and Δ'' (up to choice of variables).

In particular, it is not possible for $\otimes L$ and $/\!\!/R$ inferences to target the distinguished '$\circ$', since the right hand element is not a formula. A $\backslash\!\!\backslash R$ inference can potentially target the '$\circ$'—but then the chain containing the '$\circ$' would not be purely structural, contrary to assumption.

Since the subproofs begin synchronized (by construction), and since each step maintains synchronization, the final conclusions will be identical. Furthermore, because the same logical inferences are executed in the same way (that is, they manipulate the same formulas to build the same structures), and in the same order, and because structural inferences do not affect the Curry-Howard labeling, the semantic label of the final sequent of the original proof will be identical to that of the constructed proof.

Therefore we can eliminate all purely structural abstractions one by one until none remain. This concludes the proof of the abstraction lemma.

Here's an example of a derivation containing a purely structural abstraction. This derivation might be used to analyze covert scope-taking as in *Ann saw everyone*.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\mathrm{ann}\bullet(\mathrm{saw}\bullet\mathrm{DP})\vdash \mathrm{S}}}{\mathrm{ann}\bullet(\mathrm{DP}\circ_1\lambda x(\mathrm{saw}\bullet x))\vdash \mathrm{S}}\,\mathrm{RED}_1}{\mathrm{DP}\circ_3\lambda y(\mathrm{ann}\bullet(y\circ_1\lambda x(\mathrm{saw}\bullet x)))\vdash \mathrm{S}}\,\mathrm{RED}_3}{\lambda y(\mathrm{ann}\bullet(y\circ_1\lambda x(\mathrm{saw}\bullet x)))\vdash \mathrm{DP}\backslash\!\!\backslash \mathrm{S}}\,\backslash\!\!\backslash R \qquad \mathrm{S}\vdash\mathrm{S}}{\mathrm{S}/\!\!/(\mathrm{DP}\backslash\!\!\backslash\mathrm{S})\circ_2\lambda y(\mathrm{ann}\bullet(y\circ_1\lambda x(\mathrm{saw}\bullet x)))\vdash \mathrm{S}}\,/\!\!/L}{\mathrm{ann}\bullet(\mathrm{S}/\!\!/(\mathrm{DP}\backslash\!\!\backslash\mathrm{S})\circ_1\lambda x(\mathrm{saw}\bullet x))\vdash \mathrm{S}}\,\mathrm{EXP}_2}{\mathrm{ann}\bullet(\mathrm{saw}\bullet\mathrm{S}/\!\!/(\mathrm{DP}\backslash\!\!\backslash\mathrm{S}))\vdash \mathrm{S}}\,\mathrm{EXP}_1$$

The equivalent derivation but with the purely structural abstraction removed was given above in section 1.

2.3 Unit lemma

Just as we did for abstraction, we can separate the structural rule governing the unit into operations that introduce ("push") and eliminate ("pop") a unit:

$$\frac{\Sigma[\Delta] \vdash A}{\Sigma[1 \bullet \Delta] \vdash A}\ \text{LEFT PUSH} \qquad \frac{\Sigma[\Delta] \vdash A}{\Sigma[\Delta \bullet 1] \vdash A}\ \text{RIGHT PUSH}$$

$$\frac{\Sigma[1 \bullet \Delta] \vdash A}{\Sigma[\Delta] \vdash A}\ \text{LEFT POP} \qquad \frac{\Sigma[\Delta \bullet 1] \vdash A}{\Sigma[\Delta] \vdash A}\ \text{RIGHT POP}$$

The inference names are from Restall 2000:30; the choice of '1' for the unit comes from Ono 2003.

Only inferences in which the premises are more complex than the conclusion (i.e., the pop rules) threaten decidability, so we can concentrate on them.

Lemma (a bounded number of pops suffices): given an arbitrary derivation in NL_λ, there is an equivalent derivation (same conclusion, same Curry-Howard labeling) in which the number of pop inferences is bounded by the length of the final conclusion.

The argument will go like this. Pop inferences can be pushed upwards until they hit a blocking inference, where the inferences that are potential blockers include expansion, $\backslash L$, $\times R$, $/L$, and push. A pop blocked by a push cancels out, and can be removed from the derivation without harm. An inference can block at most one pop (except for one corner case in which a single inference blocks two pops, see below). The number of logical inferences is bounded thanks to the subformula property, and the number of abstractions is bounded by the abstraction lemma. If follows that the number of pops is also bounded by length of the final sequent.

Proof of the unit lemma: Let p be an arbitrary derivation. In view of cut admissibility, we can assume that p is cut-free. Focus on an instance of pop, and on the inference i immediately above it. In most instances, the order of pop and i can be exchanged without affecting the proof (same final conclusion, same Curry-Howard labeling). But there are a limited number of situations in which the exchange cannot occur. Here is a complete list:

- Pop blocked by an $\backslash L$, $\times R$, or $/L$ inference:

$$\frac{\dfrac{1 \vdash A \quad \Sigma[B] \vdash C}{\Sigma[1 \bullet A\backslash B] \vdash C}\backslash L}{\Sigma[A\backslash B] \vdash C}\ \text{LEFT POP} \qquad \frac{\dfrac{1 \vdash A \quad \Sigma[B] \vdash C}{\Sigma[B/A \bullet 1] \vdash C}/L}{\Sigma[B/A] \vdash C}\ \text{RIGHT POP}$$

$$\frac{\dfrac{1 \vdash A \quad \Delta \vdash B}{1 \bullet \Delta \vdash A \times B}\times R}{\Delta \vdash A \times B}\ \text{LEFT POP} \qquad \frac{\dfrac{\Delta \vdash A \quad 1 \vdash B}{\Delta \bullet 1 \vdash A \times B}\times R}{\Delta \vdash A \times B}\ \text{RIGHT POP}$$

- Pop blocked by an expansion inference (four subcases):

$$\dfrac{\dfrac{\Sigma[1 \circ \lambda x \Gamma[x \bullet \Delta]] \vdash A}{\Sigma[\Gamma[1 \bullet \Delta]] \vdash A}\,\text{EXP}}{\Sigma[\Gamma[\Delta]] \vdash A}\,\text{LEFT POP} \qquad \dfrac{\dfrac{\Sigma[1 \circ \lambda x \Gamma[\Delta \bullet x]] \vdash A}{\Sigma[\Gamma[\Delta \bullet 1]] \vdash A}\,\text{EXP}}{\Sigma[\Gamma[\Delta]] \vdash A}\,\text{RIGHT POP}$$

$$\dfrac{\dfrac{\Sigma[\Delta \circ \lambda x \Gamma[1 \bullet x]] \vdash A}{\Sigma[\Gamma[1 \bullet \Delta]] \vdash A}\,\text{EXP}}{\Sigma[\Gamma[\Delta]] \vdash A}\,\text{LEFT POP} \qquad \dfrac{\dfrac{\Sigma[\Delta \circ \lambda x \Gamma[x \bullet 1]] \vdash A}{\Sigma[\Gamma[\Delta \bullet 1]] \vdash A}\,\text{EXP}}{\Sigma[\Gamma[\Delta]] \vdash A}\,\text{RIGHT POP}$$

  The top two cases, in which it is the unit that is abstracted, correspond to using the unit as a trace for syntactic movement, as illustrated in section 1. The bottom two cases, in which it is the complement of the unit that is abstracted, correspond to the analysis of sprouting in sluicing (see Barker 2013:section 4.3 or Barker and Shan 2014:182 for an analysis of the sprouting sluice in *John left, but I don't know when*; the unit serves as a silent temporal adverbial modifying *left*).

  Note that these cases can overlap if $\Delta = 1$:

$$\dfrac{\dfrac{\dfrac{1 \circ \lambda x \Gamma[(x \bullet 1) \bullet \Pi] \vdash A}{\Gamma[(1 \bullet 1) \bullet \Pi] \vdash A}\,\text{EXP}}{\Gamma[1 \bullet \Pi] \vdash A}\,\text{LEFT POP}_2}{\Gamma[\Pi] \vdash A}\,\text{LEFT POP}_1 \qquad \dfrac{\dfrac{\dfrac{1 \circ \lambda x \Gamma[(1 \bullet x) \bullet \Pi] \vdash A}{\Gamma[(1 \bullet 1) \bullet \Pi] \vdash A}\,\text{EXP}}{\Gamma[1 \bullet \Pi] \vdash A}\,\text{LEFT POP}_2}{\Gamma[\Pi] \vdash A}\,\text{LEFT POP}_1$$

  The left-hand (right-hand) derivation matches both of the schemas in which the variable x occupies the leftmost (rightmost) position. In this one kind of situation, a single expansion inference can simultaneously block two pop inferences. We can exchange the order of the pop inferences with each other in order to select which inference is adjacent to the expansion inference; but neither one can hop over the expansion inference, so both pop inferences remain trapped beneath the expansion. In all other situations, an inference can block at most one pop inference. There are variants with Π on the left hand side of the units.

- Pop blocked by push (two subcases):

$$\dfrac{\dfrac{\Sigma[\Delta] \vdash A}{\Sigma[1 \bullet \Delta] \vdash A}\,\text{LEFT PUSH}}{\Sigma[\Delta] \vdash A}\,\text{LEFT POP} \qquad \dfrac{\dfrac{\Sigma[\Delta] \vdash A}{\Sigma[\Delta \bullet 1] \vdash A}\,\text{RIGHT PUSH}}{\Sigma[\Delta] \vdash A}\,\text{RIGHT POP}$$

  Since these proof fragments begin and end with the same sequent (with the same Curry-Howard label), they can be deleted without affecting the proof.

A pop inference is never blocked by another pop inference, although it might initially seem as if it could be:

$$\dfrac{\dfrac{\Sigma[(1_1 \bullet 1_2) \bullet \Delta] \vdash A}{\Sigma[1_1 \bullet \Delta] \vdash A}\,\text{RIGHT POP}_2}{\Sigma[\Delta] \vdash A}\,\text{LEFT POP}_1 \not\equiv \dfrac{\dfrac{\Sigma[\Delta \bullet (1_1 \bullet 1_2)] \vdash A}{\Sigma[\Delta \bullet 1_2] \vdash A}\,\text{LEFT POP}_1}{\Sigma[\Delta] \vdash A}\,\text{RIGHT POP}_2$$

At first glance, it appears that the order of these left pop and right pop inferences can't be exchanged. However, the problematic pair of inferences is equivalent to a pair that clearly can be safely swapped:

$$\dfrac{\dfrac{\Sigma[(1_1 \bullet 1_2) \bullet \Delta] \vdash A}{\Sigma[1_1 \bullet \Delta] \vdash A}\,\text{RIGHT POP}_2}{\Sigma[\Delta] \vdash A}\,\text{LEFT POP}_1 \equiv \dfrac{\dfrac{\Sigma[(1_2 \bullet 1_1) \bullet \Delta] \vdash A}{\Sigma[1_1 \bullet \Delta] \vdash A}\,\text{LEFT POP}_2}{\Sigma[\Delta] \vdash A}\,\text{LEFT POP}_1$$

By the same kind of reasoning, a right push can cancel with a left pop when the target has the form $\Sigma[1 \bullet 1]$.

With these observations in hand, we can push pop inferences as far upwards in the proof as possible. If a pop is blocked, it will get trapped immediately beneath the inference that blocks it. If the blocking inference is a push inference, the two inferences cancel out, and we can remove them. The net result is that as long as there is a bounded number of $\backslash L$, $/L$, and expansion inferences, there will be a bounded number of pops. This concludes the proof of the unit lemma.

Unlike abstraction, there are situations in which a purely structural unit—that is, a unit that is both introduced and eliminated via structural rule—cannot be eliminated. The analysis of sprouting in Barker 2013 (reproduced in Barker and Shan 2014, chapter 16) is an example. In that analysis, the introduction of a unit serves the role of a silent modifier that allows remnant movement to provide an antecedent for the sluice gap.

2.4 A decision algorithm

In order to complete the proof of decidability, it is necessary to exhibit an algorithm that is guaranteed to either find a derivation if one exists—or else to guarantee that no derivation is possible—in an amount of time bounded by the size of the sequent to be proved. Given the abstraction lemma and the unit lemma, a very simple algorithm presents itself: associate each sequent s with a budget k of expansion inferences, where k is the number of '$\backslash\!\backslash$'s, '$/\!/$'s, and '$\otimes$'s in s. Conduct the normal Gentzen-style proof search, positing expansions freely at every stage, but only as long as the number of expansions does not exceed the budget for the subproof in question. Since there are a finite number of ways to apply the expansion inference, this search is guaranteed to terminate.

Similar reasoning applies to pop inferences for the unit, except that the budget for pop inferences is bounded above by $(2 * k)$ + the number of instances of $\backslash L$, $\times R$, and $/L$ in s.

Although simple, this algorithm is crude. Many additional optimizations are possible. However, a detailed analysis of efficient parsing for NL_λ will have to wait for a different occasion.

2.5 Expressive power

Because NL_λ is a conservative extension of NL (Barker and Shan 2014 chapter 17), and NL generates the context free languages, NL_λ generates at least the context free languages. The lexicon $a = t/\!/(t\backslash\!\backslash B)$, $b = B/\!/(t\backslash\!\backslash C)$, ..., $e = E/\!/(t\backslash\!\backslash t)$ generates a non-context free indexed language, namely, the closure under permutation of $a^n b^n c^n d^n e^n$. Because NL_λ is a polynominal-parsable formalism whose languages properly includes the context free languages, it counts as a mildly contex

free formalism. It is possible to embed head grammars in NL_λ: encode the headed string $\alpha\hat{\beta}\gamma$ as $\lambda x.\alpha \bullet \beta \bullet x \bullet \gamma$ (with any bracketing), encode wrap(A,B) as $B \circ A$, and encode concatenation as, e.g., $c_{2,0}(A,B) = \lambda y((y \circ A) \bullet (1 \circ B))$. Since linear indexed grammars, tree adjoining grammars, and combinatory categorial grammars are weakly equivalent to head grammars (Vijay-Shankir and Weir 1994), NL_λ can generate any of the languages these formalisms can generate.

3 Polymorphism and an interpolation theorem

In many Lambek systems, there is a simple correspondence between structures and formulas (see, e.g., Buszkowski 2010:13) that maps $\bullet$ to $\times$, $\circ$ to $\otimes$, and 1 to t. Then anything that can be proven by a structure can be proven by replacing that structure with its corresponding formula: just use $\times L$, $\otimes L$, and tL to reconstruct the original structure, and continue with the original proof.

Extending this strategy to cover abstraction is not simple. The reason is that there is no single formula that can behave in all circumstances like a given lambda structure. Lambda structures guarantee a connection between the type of their sibling argument and the type of the resulting structure, without fixing in advance what the transmitted type is. For instance, $A \circ \lambda xx \vdash A$ for all A. It follows that $\lambda xx \vdash A \backslash\!\backslash A$, so all formulas of the form $A \backslash\!\backslash A$ have an equal claim to represent part of the provability potential of λxx. An in fact, this polymorphism is an essential part of how the logic works in order to accomplish useful linguistic work, since the structure that undergoes abstraction in a reduction inference when a lambda is introduced can differ arbitrarily from the structure that undergoes expansion when that same lambda is eliminated.

However, *when a lambda structure is embedded in a specific proof*, we *can* find a formula that accurately tracks the role the structure plays in that proof, much in the same way that a type application (an 'instantiation') in System F applies a polymorphic function to a specific type.

The following theorem will justify this claim and make it precise. The result will also be used at several points in the completeness proof below. This is the NL_λ version of a lemma proved by Jäger 2005 for $\mathrm{NL}\diamond$. Buszkowski 2005, 2010 generalizes the lemma to a larger class of Lambek logics, and names it "interpolation". Barker 2007 discusses at length a closely related notion that he calls "disclosure" in the context of Lambek grammars that handle scope-taking (and therefore, implicitly, movement).

Interpolation is the inverse of cut. In the form of a slogan: every structure can be cut.

Theorem 2 ("interpolation"): Given $\Sigma[\Delta] \vdash A$, there is a formula δ such that $\Delta \vdash \delta$ and $\Sigma[\delta] \vdash A$.

I'll use the term 'structural element' to cover structures and parts of structures (e.g., $\mathrm{DP} \bullet x$ is a structural element that is part of the complete structure $\lambda x(\mathrm{S} \bullet (\mathrm{DP} \bullet x))$). I'll write "$\delta : \Delta$" to associate a structural element Δ with the formula δ. Strictly speaking, the theorem only applies when Δ is a complete structure. The proof will extend the guarantee provided by the theorem to certain structural elements that are not complete structures in the following way: given an NL_λ proof of $\Sigma[\lambda x\Gamma[\Pi[x]]] \vdash A$, the proof of the theorem will associate the structural element $\Pi[x]$ with a formula $\delta \backslash\!\backslash \pi$ such that $\lambda x\Pi[x] \vdash \delta \backslash\!\backslash \pi$, and $\Sigma[\lambda x\Gamma[x \circ \delta \backslash\!\backslash \pi]] \vdash A$.

Note that the guarantee provided by the theorem is recursive in the following sense. If we have a proof of $\Sigma[\Gamma[\Delta]] \vdash A$, the theorem provides a formula γ such that $\Gamma[\Delta] \vdash \gamma$ and $\Sigma[\gamma] \vdash A$. But given the existence of a proof of $\Gamma[\Delta] \vdash \gamma$, there must also be a δ such that $\Delta \vdash \delta$ and $\Gamma[\delta] \vdash \gamma$, and so on.

The proof of theorem 2 proceeds by induction on the complexity of the proof of $\Sigma[\Delta] \vdash A$. Clearly the theorem holds for axiom instances $A{:}A \vdash A$. Assume that the theorem holds recursively (in the sense just described) for each premise. We must demonstrate that it holds for the conclusion as well.

($\backslash L$): Assume we have as premises $\Gamma \vdash A$ and $\Sigma[B] \vdash C$. Then we can annotate the conclusion sequent derived by $\backslash L$ as $\Sigma[B{:}(A{:}\Gamma \bullet A\backslash B{:}A\backslash B)] \vdash C$. Clearly the formulas associated with the structures Γ, $A\backslash B$, and $\Gamma \bullet A\backslash B$ satisfy the theorem. Structural elements that are part of Γ satisfy the theorem by virtue of the inductive hypothesis combined with a cut on A; the remaining structural elements satisfy the theorem by virtue of the inductive hypothesis combined with a cut on B.

($\times R$): Assume we have as premises $\Sigma \vdash A$ and $\Delta \vdash B$. Then we can annotate the conclusion sequent derived by $\times R$ as $A{:}\Sigma \bullet B{:}\Delta \vdash A \times B$. The theorem holds for this sequent by virtue of the inductive hypothesis possibly combined with a cut on either A or B.

(Reduction): Assume we have as premise $\Sigma[\gamma{:}\Gamma[\pi{:}\Pi[\delta{:}\Delta]]] \vdash A$. Then we can annotate the conclusion sequent derived by reduction as $\Sigma[\delta{:}\Delta \circ \delta \backslash\!\backslash \gamma{:}\lambda x\Gamma[\delta \backslash\!\backslash \pi{:}\Pi[x]]] \vdash A$. For any complete structure in $\lambda x\Gamma[\Pi[x]]$, applying expansion to the conclusion recreates the premise, and so the inductive hypothesis confirms the requirements of the theorem for that structure. For structural elements of the form $\delta \backslash\!\backslash \pi{:}\Pi[x]$, we first need to prove $\lambda x\Pi[x] \vdash \delta \backslash\!\backslash \pi$, which follows from $\backslash\!\backslash R$, reduction, and a recursive application of the inductive hypothesis. We also need to prove $\Sigma[\Delta \circ \lambda x\Gamma[x \circ \delta \backslash\!\backslash \pi]] \vdash A$, which follows from reduction, $\backslash\!\backslash L$, and a recursive application of the inductive hypothesis.

(Expansion): For the same reason that expansion was the main worry for the decidability proof, it is the most difficult case for theorem 2: expansion is the only inference that destroys information, since the premise contains more connectives than the conclusion. Not surprisingly, it will require careful reasoning to show that the theorem holds in the presence of expansion, since we need to keep track of the relevant derivational history in order to find the appropriate formula for structural elements in the conclusion of the expansion inference.

Assume we have as premise $\Sigma[\omega{:}\Omega \circ \delta \backslash\!\backslash \gamma{:}\lambda x\Gamma[\delta \backslash\!\backslash \pi{:}\Pi[x]]] \vdash A$. Then we can annotate the conclusion sequent derived by expansion as $\Sigma[\omega \otimes (\delta \backslash\!\backslash \gamma){:}\Gamma[\omega \otimes (\delta \backslash\!\backslash \pi){:}\Pi[\Omega]]] \vdash A$. We can prove $\Sigma[\omega \otimes (\delta \backslash\!\backslash \gamma)] \vdash A$ by $\otimes L$ and the inductive hypothesis. In addition, from $\Omega \vdash \omega$ and $\lambda x\Gamma[\Pi[x]] \vdash \delta \backslash\!\backslash \gamma$ (both given by the inductive hypothesis), $\otimes R$, and expansion, we can prove $\Gamma[\Pi[\Omega]] \vdash \omega \otimes (\delta \backslash\!\backslash \gamma)$.

For any complete structure inside of $\lambda x\Gamma[\Pi[x]]$, a reduction derives the premise sequent, and so the inductive hypothesis confirms the requirements of the theorem for that structure.

For any structure $\omega \otimes (\delta \backslash\!\backslash \pi){:}\Pi[\Omega]$ in the conclusion sequent, we first need to prove $\Pi[\Omega] \vdash \omega \otimes (\delta \backslash\!\backslash \pi)$. Using expansion and $\otimes R$, this follows from the fact that $\lambda x\Pi[x] \vdash \delta \backslash\!\backslash \pi$, which is given by the inductive hypothesis. We also need to prove that $\Sigma[\Gamma[\omega \otimes (\delta \backslash\!\backslash \pi)]] \vdash A$. Since we already have $\Sigma[\omega \otimes (\delta \backslash\!\backslash \gamma)] \vdash A$, it will suffice to prove that $\Gamma[\omega \otimes (\delta \backslash\!\backslash \pi)] \vdash \omega \otimes \delta \backslash\!\backslash \gamma$. By the inductive hypothesis, we have $\lambda x\Gamma[x \circ \delta \backslash\!\backslash \pi] \vdash \delta \backslash\!\backslash \gamma$. The desired conclusion follows from the axiom $\omega \vdash \omega$, $\otimes R$, expansion, and $\otimes L$.

Finding a suitable correspondence for the other logical rules is straightforward.

This completes the proof of theorem 2.

Example. This example tracks the formula assigned to the underlined structural element

through a reduction and an expansion.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{B{:}(B/A \bullet A) \bullet B\backslash C \vdash C}}{A \circ \lambda x(A\backslash\!\backslash B{:}(B/A \bullet x) \bullet B\backslash C) \vdash C}\,\text{RED}}{\lambda x(A\backslash\!\backslash B{:}(B/A \bullet x) \bullet B\backslash C) \vdash A\backslash\!\backslash C}\,\backslash\!\backslash R \qquad D \vdash D}{D /\!\!/ (A\backslash\!\backslash C) \circ \lambda x(A\backslash\!\backslash B{:}(B/A \bullet x) \bullet B\backslash C) \vdash D}\,/\!\!/ L}{(D /\!\!/ (A\backslash\!\backslash C))\otimes(A\backslash\!\backslash B){:}(B/A \bullet D /\!\!/ (A\backslash\!\backslash C)) \bullet B\backslash C \vdash D}\,\text{EXP}$$

We have $B/A \bullet D /\!\!/ (A\backslash\!\backslash C) \vdash (D /\!\!/ (A\backslash\!\backslash C))\otimes(A\backslash\!\backslash B)$ and $(D /\!\!/ (A\backslash\!\backslash C))\otimes(A\backslash\!\backslash B) \bullet B\backslash C \vdash D$, as required by the theorem.

Jäger uses interpolation to prove that NL$\diamond$ recognizes only context-free languages. However, his argument depends on proving that the number of logical connectives in the interpolated formula does not exceed the maximum number of connectives in the formulas in the target sequent. The example violates that requirement, since $(D /\!\!/ (A\backslash\!\backslash C))\otimes(A\backslash\!\backslash B)$ has four logical connectives, compared to a maximum of two for the formulas in the target sequent. This is not surprising given that NL$_\lambda$ can recognize non-context-free languages.

4 A sound and complete semantics for NL$_\lambda$

Because the abstraction postulate has an unusual form, it may not be obvious how to adapt the standard techniques for proving soundness and completeness. So it is worthwhile proving the following fact:

Theorem 3 ("soundness and completeness"): NL$_\lambda$ is sound and complete with respect to the standard class of relational frames.

Since the abstraction postulate relates positions separated by an unbounded amount of structure, many of the arguments below will proceed by induction on structural complexity.

A *frame* $\mathscr{F}$ for NL$_\lambda$ consists of a set of points P, a partial order '$\leq$' on P, and two 3-place accessibility relations, $R^\bullet$ and R°, one for each mode.

The accessibility relations can be any three-place relation over P, as long as they satisfy the following requirement on the interaction with the partial ordering $\leq$: for all $abcdef$ with $d \leq a$, $e \leq b$ and $c \leq f$, $R^\bullet abc \rightarrow R^\bullet def$, and likewise for R°. That is, the accessibility relations must be downward monotonic with respect to each of their first two arguments, and upward monotonic with respect to their third argument. This is what Restall 2000:240 calls a 'plump' accessibility relation. For many frames, the partial order will be the identity relation, in which case the monotonicity requirements are trivially satisfied. For the completeness proof below, however, we'll construct a canonical model in which the partial order corresponds to the subset relation.

A *model* $\mathscr{M}$ for NL$_\lambda$ is a frame plus an evaluation relation $\Vdash$ that relates points to formulas and to structures. For atomic formulas, the only restriction on the evaluation relation is that it is upward monotonic with respect to the partial order $\leq$. That is, if $p \Vdash A$ for any atomic formula A, and $p \leq q$, then $q \Vdash A$. In addition, the evaluation relation must obey the following semantic

constraints on complex formulas and on structures:

$$a \Vdash C/B \text{ iff } \forall bc.(R^\bullet abc \wedge b \Vdash B) \rightarrow c \Vdash C$$
$$b \Vdash A\backslash C \text{ iff } \forall ac.(R^\bullet abc \wedge a \Vdash A) \rightarrow c \Vdash C$$
$$c \Vdash A \times B \text{ iff } \exists ab.R^\bullet abc \wedge a \Vdash A \wedge b \Vdash B$$

$$a \Vdash C/\!\!/B \text{ iff } \forall bc.(R^\circ abc \wedge b \Vdash B) \rightarrow c \Vdash C$$
$$b \Vdash A\backslash\!\!\backslash C \text{ iff } \forall ac.(R^\circ abc \wedge a \Vdash A) \rightarrow c \Vdash C$$
$$c \Vdash A \otimes B \text{ iff } \exists ab.R^\circ abc \wedge a \Vdash A \wedge b \Vdash B$$

$$c \Vdash \Sigma \bullet \Delta \text{ iff } \exists ab.R^\bullet abc \wedge a \Vdash \Sigma \wedge b \Vdash \Delta$$
$$c \Vdash \Sigma \circ \Delta \text{ iff } \exists ab.R^\circ abc \wedge a \Vdash \Sigma \wedge b \Vdash \Delta$$
$$i \Vdash 1 \text{ iff } i \Vdash t$$

$$b \Vdash \lambda xx \text{ iff } \forall ac\Delta.(R^\circ abc \wedge a \Vdash \Delta) \rightarrow c \Vdash \Delta \quad \text{(I)}$$
$$b \Vdash \lambda x_n \ldots x_1 x_0(\Pi[x_0] \bullet \Gamma) \text{ iff } b \Vdash \lambda x_n \ldots x_1(\lambda x_0 \Pi[x_0] \circ (\Gamma \circ \lambda zyx((x \circ y) \bullet z))) \quad \text{(B}^\bullet\text{)}$$
$$b \Vdash \lambda x_n \ldots x_1 x_0(\Pi[x_0] \circ \Gamma) \text{ iff } b \Vdash \lambda x_n \ldots x_1(\lambda x_0 \Pi[x_0] \circ (\Gamma \circ \lambda zyx((x \circ y) \circ z))) \quad \text{(B}^\circ\text{)}$$
$$b \Vdash \lambda x_n \ldots x_1 x_0(\Gamma \bullet \Pi[x_0]) \text{ iff } b \Vdash \lambda x_n \ldots x_1(\lambda x_0 \Pi[x_0] \circ (\Gamma \circ \lambda zyx(z \bullet (x \circ y)))) \quad \text{(C}^\bullet\text{)}$$
$$b \Vdash \lambda x_n \ldots x_1 x_0(\Gamma \circ \Pi[x_0]) \text{ iff } b \Vdash \lambda x_n \ldots x_1(\lambda x_0 \Pi[x_0] \circ (\Gamma \circ \lambda zyx(z \circ (x \circ y)))) \quad \text{(C}^\circ\text{)}$$

The clauses for the logical connectives are standard (see, e.g., Moortgat 1997 or Restall 2000). The second to last stanza extends the evaluation relation to structures in the natural and usual way. Only the final stanza has news value, and gives the semantics for lambda abstracts. Because lambda structures are created linear (i.e., there is exactly one occurrence in the body of the structure of the variable bound by each lambda), and since no inference rule changes a linear structure into a non-linear one, these semantic clauses cover all derivable structures.

In order for the semantics to accurately model the structural postulates, it is necessary to constrain the accessibility relations $R^\bullet$ and R° in every frame.

$$\forall ab.a \leq b \leftrightarrow (\exists i.R^\bullet iab \wedge i \Vdash t) \quad \text{(left unit)}$$
$$\forall ab.a \leq b \leftrightarrow (\exists i.R^\bullet aib \wedge i \Vdash t) \quad \text{(right u't)}$$
$$\forall ab.a \leq b \leftrightarrow (\exists i.R^\circ aib \wedge i \Vdash \lambda xx) \quad \text{(I)}$$
$$\forall abcd.(\exists e.R^\circ abe \wedge R^\bullet ecd) \leftrightarrow (\exists fgh.R^\circ cfg \wedge R^\circ bgh \wedge R^\circ ahd \wedge f \Vdash \lambda zyx((x \circ y) \bullet z)) \quad \text{(B}^\bullet\text{)}$$
$$\forall abcd.(\exists e.R^\circ abe \wedge R^\circ ecd) \leftrightarrow (\exists fgh.R^\circ cfg \wedge R^\circ bgh \wedge R^\circ ahd \wedge f \Vdash \lambda zyx((x \circ y) \circ z)) \quad \text{(B}^\circ\text{)}$$
$$\forall abcd.(\exists e.R^\circ abe \wedge R^\bullet ced) \leftrightarrow (\exists fgh.R^\circ cfg \wedge R^\circ bgh \wedge R^\circ ahd \wedge f \Vdash \lambda zyx(z \bullet (x \circ y))) \quad \text{(C}^\bullet\text{)}$$
$$\forall abcd.(\exists e.R^\circ abe \wedge R^\circ ced) \leftrightarrow (\exists fgh.R^\circ cfg \wedge R^\circ bgh \wedge R^\circ ahd \wedge f \Vdash \lambda zyx(z \circ (x \circ y))) \quad \text{(C}^\circ\text{)}$$

These frame conditions are similar in spirit and many details (notably with respect to the names I, B, and C) to those proposed in Barker 2007a and discussed in chapter 17 of Barker and Shan 2014.

Some standard vocabulary: let '$[\![X]\!]$' be the set of all points x such that $x \Vdash X$. We say that a sequent $X \Vdash A$ is *valid* if $[\![X]\!] \subseteq [\![A]\!]$ in every frame. A logic is *sound* with respect to a class of frames just in case every provable sequent is valid, and it is *complete* just in case every valid sequent is provable.

4.1 Soundness

As usual, soundness is proved by induction on proof length. Axiom instances are trivially sound, and it is easy to show for the logical inference rules that whenever the premises are sound, the conclusion is also sound. We only need to worry here about the structural inferences.

For the left unit, we must show that $q \Vdash \Delta$ iff $q \Vdash 1 \bullet \Delta$. In the left to right direction, assume that $q \Vdash \Delta$. Then $q \Vdash 1 \bullet \Delta$ iff there exists i and p such that $i \Vdash 1$, $p \Vdash \Delta$, and $R^\bullet ipq$. According to the evaluation condition for the unit, $i \Vdash 1$ iff $i \Vdash t$. Frame condition (left unit) guarantees the existence of a point i such that $i \Vdash t$ and $R^\bullet ipq$ iff $p \leq q$. If we choose $p = q$, we have $i \Vdash 1$ and $R^\bullet ipq$, so $q \Vdash 1 \bullet \Delta$.

In the right to left direction, assume $q \Vdash 1 \bullet \Delta$. Then there are i and p such that $i \Vdash t$, $p \Vdash \Delta$, and $R^\bullet ipq$. But by the frame condition, the existence of such an i entails $p \leq q$. This is enough to guarantee that $q \Vdash \Delta$, in view of the following lemma: $[\![\Gamma]\!]$ is upward monotonic with respect to $\leq$, for all structures Γ. That is, if $p \Vdash \Gamma$ and $p \leq q$, then $q \Vdash \Gamma$.

Here's a sketch of a proof (see also Restall 2000:242). The lemma holds for atomic formulas by stipulation. If $\Gamma = \Sigma \bullet \Delta$, there exist a and b such that $a \Vdash \Sigma$, $b \Vdash \Delta$, and $R^\bullet abp$. But since $R^\bullet$ is plump, it is upward monotonic in its third argument, so it follows that $R^\bullet abq$. Similarly for structures built from $\circ$, $\times$, or $\otimes$. If $\Gamma = A \backslash C$, then $p \Vdash \Gamma$ iff for all a and c, $R^\bullet apc$ and $a \Vdash A$ guarantees $c \Vdash C$. But since $R^\bullet aqc \rightarrow R^\bullet apc$, it follows that $p \Vdash \Gamma$ only if $q \Vdash \Gamma$. Similarly for structures built from $/$, $\backslash\!\backslash$, and $/\!/$. If $\Gamma = \lambda xx$, then $p \Vdash \Gamma$ iff for all a, c, and Δ, $R^\circ apc$ and $a \Vdash \Delta$ guarantees $c \Vdash \Delta$. But again, since $R^\circ aqc \rightarrow R^\circ apc$, it follows that $p \Vdash \Gamma$ only if $q \Vdash \Gamma$. Finally, if $\Gamma = \lambda x \Sigma[x]$ where $\Sigma[\ ]$ is not empty, the applicable evaluation clause establishes an equivalence with a structure whose top level connective is $\circ$, for which we have already shown that the lemma holds.

A similar argument holds for the right unit.

For abstraction, we need to show that for all points c and all structures Δ and $\Sigma[\Delta]$, $c \Vdash \Sigma[\Delta]$ iff $c \Vdash \Delta \circ \lambda x \Sigma[x]$. The proof proceeds by induction on the structural complexity of $\lambda x \Sigma[x]$.

For the base case, $\lambda x \Sigma[x] = \lambda xx$. We need to show that for all points q, $q \Vdash \Delta$ iff $q \Vdash \Delta \circ \lambda xx$. Since λxx functions as a right unit for the $\circ$ mode, the argument just given for the $\bullet$ mode units applies here with minor modification.

For the inductive case, consider first the subcase in which $\Sigma[\Delta]$ has the form $\Pi[\Delta] \bullet \Gamma$. We need to show that for all points c, $c \Vdash \Pi[\Delta] \bullet \Gamma$ iff $c \Vdash \Delta \circ \lambda x(\Pi[x] \bullet \Gamma)$. We reason as follows:

$c \Vdash \Pi[\Delta] \bullet \Gamma$	
iff $c \Vdash (\Delta \circ \lambda x \Pi[x]) \bullet \Gamma$	(by the inductive hypothesis)
iff $c \Vdash \Delta \circ (\lambda x \Pi[x] \circ (\Gamma \circ \lambda zyx((x \circ y) \bullet z)))$	(by frame condition $B^\bullet$)
iff $c \Vdash \Delta \circ \lambda x(\Pi[x] \bullet \Gamma)$	(by the semantic clause for $B^\bullet$)

Similar arguments cover subcases involving (B°), ($C^\bullet$), and (C°). For subcases in which the starting configuration is a lambda structure, we apply the relevant semantic clauses as many times as required in order to arrive at a $\circ$ structure at the top level. For instance,

$$\begin{aligned} & c \Vdash \lambda y(\Pi[\Delta] \bullet \Gamma[y]) \\ \text{iff } & c \Vdash \lambda y \Gamma[y] \circ (\Pi[\Delta] \circ \lambda zyx(y \circ (x \circ z))) \qquad \text{(by the semantic clause for } C^\bullet) \end{aligned}$$

The second line is an instance of the (C°) subcase. In general, abstraction from a lambda structure can always be reduced to a structure with $\circ$ as the top-level connective. So the inductive case holds, and the logic is sound with respect to the semantics.

4.2 Completeness

A logic is (weakly) complete with respect to a class of frames just in case whenever an inference is valid, it is provable, i.e., whenever $[\![X]\!] \subseteq [\![A]\!]$ for all models in all frames, $X \vdash A$ is provable in the logic. I will prove completeness in the usual way, by constructing a 'canonical' frame and model that only validates provable sequents.

In some completeness proofs for Lambek-style logics, points in the canonical point set are taken to be formulas of the logic. This strategy won't work here, for the same reason given above in section 3, namely, that the polymorphism of lambda structures prevents using any single formula to represent an abstract.

However, allowing points to be *sets* of formulas will suffice, following the general lines of the parametric proof of completeness for substructural logics given in Restall 2000 chapter 11.

Therefore let P, the set of points in the canonical frame, contain all non-empty sets of formulas that are closed under provability. That is, if we have some formula $A \in p$ for some point p, and $A \vdash B$, then $B \in p$. Then the subset relation $\subseteq$ will serve as the partial order $\leq$.

The accessibility relations for the canonical frame are defined as follows for all points a, b, and c, and for all formulas A, and B:

$$\begin{aligned} R^\bullet abc \text{ iff } \forall AB.(A \in a \wedge B \in b) \rightarrow A \times B \in c \\ R^\circ abc \text{ iff } \forall AB.(A \in a \wedge B \in b) \rightarrow A \otimes B \in c \end{aligned}$$

Clearly these accessibility relations are plump with respect to $\subseteq$.

Adding a canonical evaluation relation '$\Vdash$' defines the canonical model. For all points p and structures Δ,

$$p \Vdash \Delta \text{ iff } \{A \mid \Delta \vdash A \text{ is provable in } \mathrm{NL}_\lambda\} \subseteq p$$

In particular, when Δ is a single formula A, given that points are closed under provability, this definition reduces to

$$p \Vdash A \text{ iff } A \in p$$

The accessibility relations and the evaluation relation work together to model provability in the logic.

Constraints on the evaluation relation. We need to show that the canonical evaluation relation $\Vdash$ satisfies the general constraints on evaluation relations given above.

For atomic propositions A, $\Vdash$ is clearly upward monotonic with respect to $\leq$, since if $A \in p$ and $p \subseteq q$, then $A \in q$.

($\times$): The evaluation clause for '$\times$' is repeated here:

$$c \Vdash A \times B \text{ iff } \exists ab.R^\bullet abc \wedge a \Vdash A \wedge b \Vdash B$$

In the left to right direction, assume $c \Vdash A \times B$. Choose $a = \{A' \mid A \vdash A'\}$, and $b = \{B' \mid B \vdash B'\}$. Then $R^\bullet abc$ iff $\forall A' \in a, B' \in b. A' \times B' \in c$. We know that $A \times B \in c$ by assumption, and that $A \times B \vdash A' \times B'$ (use $\times L$ and $\times R$), and since c is closed under provability, $A' \times B' \in c$. The right to left direction follows immediately from the definition of $R^\bullet$.

($\otimes$): Similar to ($\times$).

($\backslash$): The evaluation clause for '$\backslash$' is repeated here:

$$b \Vdash A \backslash C \text{ iff } \forall ac.(R^\bullet abc \wedge a \Vdash A) \rightarrow c \Vdash C$$

In the left to right direction, assume $b \Vdash A \backslash C$. Choose arbitrary a and c, and assume $R^\bullet abc$ and $a \Vdash A$. Then $A \times (A \backslash C) \in c$ by the definition of $R^\bullet$, and since $A \times (A \backslash C) \vdash C$ and c is closed under provability, $c \Vdash C$. In the right to left direction, assume the right hand proposition holds. Choose $a = \{A' \mid A \vdash A'\}$, and let $c = \{C' \mid \exists A \in a, B \in b : A \times B \vdash C'\}$. Then $R^\bullet abc$ by the construction of c, so $C \in c$ by assumption. But also by the construction of c, there must exist some $A' \in a$ and $B \in b$ such that $A' \times B \vdash C$. By $\times L$ and cut, $A \bullet B \vdash C$, and so $B \vdash A \backslash C$. Since b is closed under provability, $b \Vdash A \backslash C$.

($/$), ($\backslash\!\backslash$), ($/\!/$): Similar to ($\backslash$).

We must also check that the canonical evaluation relation extends to structures in the appropriate way.

($\bullet$): Here is the evaluation condition for $\bullet$:

$$c \Vdash \Sigma \bullet \Delta \text{ iff } \exists ab.R^\bullet abc \wedge a \Vdash \Sigma \wedge b \Vdash \Delta$$

In the left to right direction, assume $c \Vdash \Sigma \bullet \Delta$. Choose $a = \{A \mid \Sigma \vdash A\}$ and $b = \{B \mid \Delta \vdash B\}$. Since $\Sigma \bullet \Delta \vdash A \times B$ for all $A \in a$ and for all $B \in b$ (use $\times R$), we have $R^\bullet abc$.

In the right to left direction, choose a and b as before, so $a = \{A \mid \Sigma \vdash A\}$ and $b = \{B \mid \Delta \vdash B\}$, and assume that $R^\bullet abc$. Choose an arbitrary C such that $\Sigma \bullet \Delta \vdash C$. By theorem 2, there exist σ and δ such that $\Sigma \vdash \sigma$ (which means $\sigma \in a$), $\Delta \vdash \delta$ (which means $\delta \in b$), and $\sigma \times \delta \vdash C$. Since $R^\bullet abc$, $\sigma \times \delta \in c$, so $C \in c$.

($\circ$): Similar to ($\bullet$).

(1): The evaluation clause for 1 says that for all i, $i \Vdash 1$ iff $i \Vdash t$. In the left to right direction, assume that $i \Vdash 1$. Since $1 \vdash t$, and i is closed under provability, $i \Vdash t$. In the right to left direction, assume $i \Vdash t$. Then for all A such that $1 \vdash A$, it follows that $t \vdash A$ by tL, and since points are closed under provability, $i \Vdash 1$.

(λ) Last but not least, we must check that the evaluation clauses for lambda structures hold. We begin with the clause for λxx:

$$b \Vdash \lambda xx \text{ iff } \forall ac\Delta.(R^\circ abc \wedge a \Vdash \Delta) \rightarrow c \Vdash \Delta$$

In the left to right direction, assume $b \Vdash \lambda xx$, and choose arbitrary a, c, and Δ such that $R^\circ abc$ and $a \Vdash \Delta$. Then $c \Vdash \Delta \circ \lambda xx$, so $c \Vdash \Delta$.

In the right to left direction, assume the right hand condition. We need to show that for all B such that $\lambda xx \vdash B$, $B \in b$. If $\lambda xx \vdash B$, by theorem 2, there is a formula D such that $\lambda xx \vdash D \backslash\!\!\backslash D$ and $D \backslash\!\!\backslash D \vdash B$. So all we need to show at this point is that $D \backslash\!\!\backslash D \in b$.

To show this, choose $a = \{A \mid D \vdash A\}$, $c = \{C \mid A \in a, B \in b, A \otimes B \vdash C\}$, and $\Delta = D$. Then $R^\circ abc$ and $a \Vdash \Delta$, by construction, so we have $c \Vdash \Delta$, by assumption. But if $c \Vdash \Delta$, then $c \Vdash \Delta \circ \lambda xx$, so there exists a' and b' such that $a' \Vdash \Delta$, $b' \Vdash \lambda xx$, and $R^\circ a'b'c$. Since $D \in a'$, this means that $D \otimes (D \backslash\!\!\backslash D) \in c$. But by the construction of c, there exists a B' such that $D \otimes B' \vdash D \otimes (D \backslash\!\!\backslash D)$, from which it follows that $B' \vdash D \backslash\!\!\backslash (D \otimes (D \backslash\!\!\backslash D))$. Since $D \backslash\!\!\backslash (D \otimes (D \backslash\!\!\backslash D)) \vdash (D \backslash\!\!\backslash D)$ (use $\backslash\!\!\backslash R$, $\backslash\!\!\backslash L$, $\otimes L$, and $\backslash\!\!\backslash L$), by cut we have $B' \vdash D \backslash\!\!\backslash D$.

There are five semantic clauses governing abstraction, and we have just considered the first, (I). Before turning to the remaining four semantic constraints, we must establish a lemma.

Lemma ("reabstraction"): $\Sigma[\lambda x \Pi[\Gamma[x]]] \vdash A$ is provable iff $\Sigma[\lambda y \Pi[y \circ \lambda x \Gamma[x]]] \vdash A$ is.

Note that neither structure is directly provable from the other, since variables are not structures: they do not have a sensible denotation independent of their context, nor do we want to allow them to be abstracted outside of the scope of the lambda that binds them.

However, proving the lemma is straightforward. Consider the reduction inference that creates the lambda structure in question as schematized on the left:

$$\dfrac{\dfrac{\Sigma'[\Pi'[\Gamma'[\Delta]]] \vdash A'}{\Sigma'[\Delta \circ \lambda x \Pi'[\Gamma'[x]]] \vdash A'}\,\text{RED}}{\vdots}{\Sigma[\lambda x \Pi[\Gamma[x]]] \vdash A} \qquad \text{iff} \qquad \dfrac{\dfrac{\dfrac{\Sigma'[\Pi'[\Gamma'[\Delta]]] \vdash A'}{\Sigma'[\Pi'[\Delta \circ \lambda x \Gamma'[x]]] \vdash A'}\,\text{RED}}{\Sigma'[\Delta \circ \lambda y \Pi'[y \circ \lambda x \Gamma'[x]]] \vdash A'}\,\text{RED}}{\vdots}{\Sigma[\lambda y \Pi[y \circ \lambda x \Gamma[x]]] \vdash A'}$$

Relying on the reasoning from the decidability proof, the presence of the "$y \circ \lambda x$" in the right hand proof does not impede the application of any of the inferences in the continuation of the left hand proof. Nor does it enable any inferences that can't be replicated in the other direction.

This lemma expresses a sense in which abstraction is transitive: abstracting a structure and then abstracting it again is equivalent to a single abstraction that relates the starting point with the ending point. See Barker 2007b for a lengthy discussion.

With the reabstraction lemma in hand, we can address the clause for ($B^\bullet$):

$$b \Vdash \lambda x_n \ldots x_1 x_0 (\Pi[x_0] \bullet \Gamma) \text{ iff } b \Vdash \lambda x_n \ldots x_1 (\lambda x_0 \Pi[x_0] \circ (\Gamma \circ \lambda zyx((x \circ y) \bullet z)))$$

Given an arbitrary $B \in b$, we can reason as follows:

$$\begin{aligned}
&\lambda x_n \ldots x_1 x_0 (\Pi[x_0] \bullet \Gamma) \vdash B \\
&\text{iff } \lambda x_n \ldots x_1 x((x \circ \lambda x_0 \Pi[x_0]) \bullet \Gamma) \vdash B && \text{(by the reabstraction lemma)} \\
&\text{iff } \lambda x_n \ldots x_1 (\lambda x_0 \Pi[x_0] \circ (\Gamma \circ \lambda zyx((x \circ y) \bullet z))) \vdash B && \text{(expansion (twice))}
\end{aligned}$$

Thus the canonical model guarantees that the two sides of the iff in the evaluation condition prove the same set of formulas. It follows that any point that satisfies one satisfies the other.

The clauses for B°, $C^\bullet$, and C° are similar.

At this point, we have shown that the canonical evaluation relation satisfies all of the general constraints on the evaluation relation.

Frame conditions for structural rules. The canonical model also satisfies the frame conditions for the structural rules. In order to establish this fact, we need to show that there are points in the canonical model that behave under the canonical evaluation relation as required by the frame conditions.

(left unit): Here is the frame constraint for the left unit:

$$\forall ab.a \leq b \leftrightarrow (\exists i.R^\bullet iab \wedge i \Vdash t)$$

Choose arbitrary a and b. In the left to right direction, assume $a \leq b$. Let $i = \{A \,|\, t \vdash A\}$, so we have $i \Vdash t$. Then $R^\bullet iab$ iff for all $I \in i$, $A \in a$, $I \times A \in b$. Since $a \leq b$, $A \in b$, and since b is closed under provability, $t \times A \in b$ and therefore $I \times A \in b$. In the right to left direction, assume that $R^\bullet iab$. Then $I \times A \in b$ for all $I \in i$ and $A \in a$. Since $I \times A \vdash A$, and b is closed under provability, $a \subseteq b$.

(right unit), (I): Similar to (left unit).

($B^\bullet$): Here is frame condition ($B^\bullet$):

$$\forall abcd.(\exists e.R^\circ abe \wedge R^\bullet ecd) \leftrightarrow (\exists fgh.R^\circ cfg \wedge R^\circ bgh \wedge R^\circ ahd \wedge f \Vdash \lambda zyx((x \circ y) \bullet z))$$

Choose arbitrary a, b, c, and d. In the left to right direction, assume there exists e such that $R^\circ abe \wedge R^\bullet ecd$. Let e' be the closure under provability of $\{A \otimes B \,|\, A \in a, B \in b\}$. Since $R^\bullet$ is plump, it is downward monotone in its first argument, and since $e' \subseteq e$, we have $R^\bullet e'cd$. It follows that $d \supseteq \{(A \otimes B) \times C \,|\, A \in a, B \in b, C \in c\}$. Let f be the closure under provability of $\{Z \backslash\!\backslash (Y \backslash\!\backslash (X \backslash\!\backslash ((X \otimes Y) \times Z))\}$, where X, Y and Z are any formulas, and let g and h be the smallest sets that satisfy the requirements for $R^\circ cfg$ and $R^\circ bgh$ to hold. Then $R^\circ ahd$ just in case d is closed under provability and

$$d \supseteq \{A \otimes (B \otimes (C \otimes (Z \backslash\!\backslash (Y \backslash\!\backslash (X \backslash\!\backslash ((X \otimes Y) \times Z))))))\}$$

It is easy to prove that $(A \otimes B) \times C \vdash A \otimes (B \otimes (C \otimes (Z \backslash\!\backslash (Y \backslash\!\backslash (X \backslash\!\backslash ((X \otimes Y) \times Z))))))$ (use abstraction three times), and since d is closed under provability, $R^\circ ahd$.

In the right to left direction, choose f, g, and h as before. Then $R^\circ ahd$ only if

$$d \supseteq \{A \otimes (B \otimes (C \otimes (C \backslash\!\backslash (B \backslash\!\backslash (A \backslash\!\backslash ((A \otimes B) \times C))))))\}$$

To see this, just choose $X = A$, $Y = B$, and $Z = C$. Choose e' as before. It is easy to prove that $A \otimes (B \otimes (C \otimes (C \backslash\!\backslash (B \backslash\!\backslash (A \backslash\!\backslash ((A \otimes B) \times C))))))) \vdash (A \otimes B) \times C$ (use $\backslash\!\backslash L$ three times), so $R^\bullet e'cd$.

It remains only to show that $f \Vdash \lambda zyx((x \circ y) \bullet z)$. This will true just in case every K such that $\lambda zyx((x \circ y) \bullet z) \vdash K$ is also such that $K \in f$. Choose an arbitrary such K. By theorem 2, the labeling algorithm gives a formula K' such that $\lambda zyx((x \circ y) \bullet z) \vdash K'$ and $K' \vdash K$. So it will be enough to prove that $K' \in f$. By the labeling construction for reduction inferences, K' has the form $Z \backslash\!\backslash (Y \backslash\!\backslash (X \backslash\!\backslash ((X \otimes Y) \times Z)))$ for some choice of X, Y, and Z. But all formulas with this shape are in f.

(B°), ($C^\bullet$), and (C°): Similar to ($B^\bullet$).

Proof of completeness. So now we can finish the main argument. Assume $\Delta \vdash A$ is valid for some structure Δ and some formula A. Let $d = \{D \,|\, \Delta \vdash D\}$. Then $d \Vdash \Delta$. Since $\Delta \vdash A$ is valid, it follows that $d \Vdash A$. But $d \Vdash A$ iff $A \in d$. So by the construction of d, $\Delta \vdash A$ is provable in NL_λ.

5 Conclusions

NL_λ is a garden-variety Lambek-style substructural logic that has been enriched by the addition of an unconventional structural inference ("abstraction"). Despite the unusual form of the abstraction inference, the logic is well-defined. Furthermore, it is well-behaved with respect to the usual properties expected of substructural logics, notably including cut elimination, interpolation, and decidability. In addition, NL_λ is sound and complete with respect to the usual class of relational models for substructural logics.

Although the abstraction inference rule may be unconventional in comparison with standard structural rules, it closely resembles the traditional linguistic rule of Quantifier Raising, as well as the traditional notion of syntactic movement more generally. The many linguistic analyses of scope, binding, ellipsis, and movement that have been developed using NL_λ suggest that it is indeed well-suited to linguistic theorizing. I conclude that NL_λ is good candidate for the logic of scope taking and syntactic movement.

6 References

Barker, Chris. 2007a. Parasitic scope. *Linguistics and Philosophy* **30.4**:407–444.

Barker, Chris. 2007b. Direct Compositionality on Demand. In Chris Barker and Pauline Jacobson (eds). *Direct Compositionality*. Oxford University Press. 102–131.

Barker, Chris. 2013. Scopability and sluicing. *Linguistics and Philosophy* **36.3**:187–223.

Barker, Chris. 2015a. Scope. In Shalom Lappin and Chris Fox (eds). *The Handbook of Contemporary Semantics*, 2d edition. Wiley-Blackwell. 47–87.

Barker, Chris. 2015b. Scope as Syntactic Abstraction. In Tsuyoshi Murata, Koji Mineshima, and Daisuke Bekki (eds). *New Frontiers in Artificial Intelligence*. 184–199.

Barker, Chris. In prep. The logic of movement. NYU manuscript.

Barker, Chris and Chung-chieh Shan. 2014, *Continuations and Natural Language*, Oxford University Press.

Buszkowski, Wojciech. 2005. Lambek Calculus with Nonlogical Axioms. In C. Casadio, P.J. Scott, and R.A.G. Seely (eds). *Language and Grammar. Studies in Mathematical Linguistics and Natural Language*. CSLI Lecture Notes volume 168. 77–93.

Buszkowski, Wojciech. 2010. Lambek Calculus and Substructural Logics. *Linguistic Analysis*, **36**:15–48.

de Groote, Philippe. 2002. Towards abstract categorial grammars. In Proceedings of the 40th annual meeting of the Association for Computational Linguistics, 148155. San Francisco, CA: Morgan Kaufmann.

Heim, Irene, and Angelika Kratzer. 1998. *Semantics in Generative Grammar*. Blackwell, Oxford.

Jäger, Gerhard. 2005. Residuation, structural rules, and context freeness. *Journal of Logic, Language, and Information* **13.1**: 49–57.

Kubota, Yusuke. 2015. Nonconstituent Coordination in Japanese as Constituent Coordination: An Analysis in Hybrid Type-Logical Categorial Grammar. *Linguistic Inquiry* **46.1**:1–42.

Lambek, Joachim. 1958. The mathematics of sentence structure. The *American Mathematical Monthly* **65**: 154–170.

Lambek, Joachim. 1961. On the calculus of syntactic types. In *Structure of Language and Its Mathematical Aspects*, edited by R. Jakobson, 166–178. Providence: American Mathematical Society.

Lambek, Joachim. 1988. Categorial and Categorical Grammars. In Richard T. Oehrle et al. (eds). *Categorial Grammars and Natural Language Structures*. 297–317. Reidel.

May, Robert. 1977. *The Grammar of Quantification*. Ph.D. thesis, MIT. Reprinted by New York: Garland, 1991.

May, Robert. 1985. *Logical Form: Its Structure and Derivation*. MIT Press, Cambridge.

Moortgat, Michael. 1997. Categorial type logics. In Johan F. A. K. van Benthem and Alice G. B. ter Meulen (eds). *Handbook of Logic and Language*. Elsevier. 93–178.

Morrill, Glyn, Oriol Valentín, and M. Fadda. 2011. The displacement calculus. *Journal of Logic, Language and Information* **20.1**: 1–48.

Muskens, Reinhard. 2001. λ-grammars and the syntax-semantics interface. In Robert van Rooy and Martin Stokhof (eds), *Proceedings of the 13th Amsterdam Colloquium*. Institute for Logic, Language and Computation, Universiteit van Amsterdam. 150–155.

Oehrle, Richard T. 1994. Term-labeled categorial type systems. *Linguistics and Philosophy* **17.6**:633–678.

Ono, Hiroakira. 2003. Substructural Logics and Residuated Lattices—an Introduction. In V.F. Hendricks and J. Malinowski (eds). *Trends in Logic: 50 Years of Studia Logica*. 193–228.

Restall, Greg. 2000. *An Introduction to Substructural Logics*. Routledge.

Vijay-Shankir, J., and David Wier. 1994. The equivalence of four extensions of context-free grammars. *Mathematical Systems Theory* **27.6**. 511–546.