Lambek Calculus: Recognizing Power and Complexity

Makoto Kanazawa

Abstract

I survey known results and results that easily follow from known results about the recognizing power and complexity of various fragments of the Lambek calculus.

Contents

1 Complexity and recognizing power of logics **2**

2 Lambek calculus **3**

3 Multiplicative-exponential Lambek calculus **5**

4 Multiplicative-additive Lambek calculus **8**

5 Multiplicative Lambek calculus **12**

1 Complexity and recognizing power of logics

We first have to explain what we mean by the *complexity* and the *recognizing power* of a logic. To most people, the former notion should be familiar, but the latter may sound novel.

We assume that a logic K is presented in the form of a sequent calculus, which is the favorite format for substructural logics. A sequent is an expression of the form

$$A_1, \ldots, A_m \vdash B_1, \ldots, B_n$$

where $m, n \geq 0$ and $A_1, \ldots, A_m, B_1, \ldots, B_n$ are formulas.[1] We denote the set of well-formed formulas of K by Form_K and the set of sequents of K by Seq_K.

Any logic K determines the set Prov_K of *provable sequents*. Members of Prov_K are expressions made up from a finite stock of symbols, so Prov_K is a language in the sense of formal language theory. The *complexity* of K is defined to be the complexity-theoretic characteristic of this language Prov_K. We can be said to have determined the complexity of K if we have succeeded in locating Prov_K precisely in the space of complexity classes as we know it today. Thus, if we know that Prov_K is, say, PSPACE-complete, we know the complexity of K, in which case we also say that K is PSPACE-complete.

The complexity of a logic K is the property of a *single* language associated with K. In contrast, the *recognizing power* of K has to do with a *class* of languages. Given a fixed alphabet Σ, K is said to *recognize* a language $L \subseteq \Sigma^+$ if there exist some finite subset $\mathbf{A}$ of Form_K, some $B \in \mathrm{Form}_K$, and some relation $R \subseteq \Sigma \times \mathbf{A}$ such that

$$L = \{\, w \in \Sigma^+ \mid \exists \Gamma (w \mathrel{\widetilde{R}} \Gamma \wedge \Gamma \vdash B \in \mathrm{Prov}_K) \,\},$$

where $\widetilde{R} \subseteq \Sigma^* \times \mathrm{Form}_K^*$ is the extension of R to a relation between strings of symbols and strings of formulas:

$$\begin{aligned} w \mathrel{\widetilde{R}} \Gamma \Leftrightarrow\ & w = \Gamma = \epsilon \vee \\ & \exists c \in \Sigma\, \exists v \in \Sigma^*\, \exists A \in \mathrm{Form}_K\, \exists \Delta \in \mathrm{Form}_K^* \\ & (w = cv \wedge \Gamma = A\Delta \wedge c \mathrel{R} A \wedge v \mathrel{\widetilde{R}} \Delta). \end{aligned}$$

Then Rec_K is defined to be the class of all languages (over Σ) recognized by K. Note that in this definition, the empty string is disregarded, so that languages recognized consist of non-empty strings. This is for the sake of simplifying some results. Henceforth, when we talk about recognizing power, we simply say 'language' to mean 'language without the empty string'. We know the *recognizing power* of K when we can characterize Rec_K in terms of familiar notions from formal language theory, e.g., when we know that Rec_K is the class

[1] Sometimes one-sided sequents of the form

$$\vdash A_1, \ldots, A_n$$

are used, but we use the more general two-sided format.

of all context-free languages (i.e., context-free languages that do not include the empty string).

The notion of recognizing power has its origin in the Lambek calculus, where a pair $\langle R, B\rangle$ with $R \subseteq \Sigma \times \mathbf{A}$ for some finite $\mathbf{A}$ is viewed as a grammar. Yet the notion applies to any logic presented in the form of a sequent calculus. When $G = \langle R, B\rangle$, where R and B are as above, we say that G is a K-grammar, and write $\mathrm{L}(G)$ for the language $\{\, w \in \Sigma^+ \mid \exists \Gamma(w \mathrel{\widetilde{R}} \Gamma \wedge \Gamma \vdash B \in \mathrm{Prov}_K) \,\}$.

There are some properties of Rec_K that do not depend on any specific choice of K. For instance, Rec_K is always closed under *strictly alphabetic morphisms*, homomorphisms that are length-preserving. Also, if K_2 is a conservative extension of K_1, one has $\mathrm{Rec}_{K_1} \subseteq \mathrm{Rec}_{K_2}$.

Some connections between Prov_K and Rec_K are apparent. For instance, if Prov_K is recursive, then all languages in Rec_K are recursive, and likewise, if Prov_K is in NP, Rec_K is a subclass of NP. An additional interest of investigating Rec_K lies in the fact that it sometimes admits of a natural language-theoretic characterization which cannot be captured by a rough complexity-theoretic measure, and which sometimes provides valuable additional information about the logic in question.

2 Lambek calculus

We are interested in a family of logics related to the system first presented in Lambek 1958, 1961. In modern terms, they are systems of *noncommutative intuitionistic propositional linear logic*. In these systems, sequents are of the form

$$A_1, \ldots, A_m \vdash B$$

having exactly one succedent formula. Lambek also required that $m \geq 1$, but we choose not to do so here. So the antecedent of a sequent can be empty. This choice is significant in just one place.

There are six propositional connectives: $\otimes$, $\multimap$, $\circ\!\!-$, $\&$, $\oplus$, and $!$. The systems we consider here differ only in which of these connectives are used. The connectives $\otimes$, $\multimap$, and $\circ\!\!-$ are called *multiplicatives*, and $\&$ and $\oplus$ are called *additives*. The connective $!$ is the *exponential* connective. This terminology is borrowed from linear logic. The multiplicatives of the Lambek calculus are usually written $\bullet$, $\backslash$, $/$, but here we use different notations to emphasize the connection with linear logic. Some people have used these notations from linear logic for the Lambek calculus.

As in other logics, *axioms* of the Lambek calculus are sequents of the form

$$A \vdash A,$$

where A is any well-formed formula. Rules of inference associated with the different connectives are listed in Figure 1. Here as elsewhere, upper-case Greek letters stand for finite sequences of formulas and upper-case Roman letters stand for formulas. In ($\vdash !$), $!\Gamma$ stands for the result of prefixing $!$ to each formula in Γ.

$$\frac{\Gamma, A, B, \Delta \vdash C}{\Gamma, A \otimes B, \Delta \vdash C}\ (\otimes \vdash) \qquad \frac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B}\ (\vdash \otimes)$$

$$\frac{\Gamma \vdash A \quad \Delta, B, \Pi \vdash C}{\Delta, \Gamma, A \multimap B, \Pi \vdash C}\ (\multimap \vdash) \qquad \frac{A, \Gamma \vdash B}{\Gamma \vdash A \multimap B}\ (\vdash \multimap)$$

$$\frac{\Gamma \vdash A \quad \Delta, B, \Pi \vdash C}{\Delta, B \circ\!\!- A, \Gamma, \Pi \vdash C}\ (\circ\!\!- \vdash) \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash B \circ\!\!- A}\ (\vdash \circ\!\!-)$$

$$\frac{\Gamma, A, \Delta \vdash C}{\Gamma, A \,\&\, B, \Delta \vdash C}\ (\& \vdash_1) \qquad \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \,\&\, B}\ (\vdash \&)$$

$$\frac{\Gamma, B, \Delta \vdash C}{\Gamma, A \,\&\, B, \Delta \vdash C}\ (\& \vdash_2)$$

$$\frac{\Gamma, A, \Delta \vdash C \quad \Gamma, B, \Delta \vdash C}{\Gamma, A \oplus B, \Delta \vdash C}\ (\oplus \vdash) \qquad \frac{\Gamma \vdash A}{\Gamma \vdash A \oplus B}\ (\vdash \oplus_1)$$

$$\frac{\Gamma \vdash B}{\Gamma \vdash A \oplus B}\ (\vdash \oplus_2)$$

$$\frac{\Gamma, A, \Delta \vdash C}{\Gamma, !A, \Delta \vdash C}\ (! \vdash) \qquad \frac{!\Gamma \vdash C}{!\Gamma \vdash !C}\ (\vdash !)$$

$$\frac{\Gamma, \Delta \vdash C}{\Gamma, !A, \Delta \vdash C}\ \mathbf{!W} \qquad \frac{\Gamma, !A, \Delta, !A, \Pi \vdash C}{\Gamma, !A, \Delta, \Pi \vdash C}\ \mathbf{!C}_1$$

$$\frac{\Gamma, !A, \Delta, !A, \Pi \vdash C}{\Gamma, \Delta, !A, \Pi \vdash C}\ \mathbf{!C}_2$$

Figure 1: Rules of inference of the Lambek calculus.

The Lambek calculus has just one structural rule, namely Cut, which is eliminable.

$$\frac{\Gamma \vdash A \quad \Delta, A, \Pi \vdash C}{\Delta, \Gamma, \Pi \vdash C}\ \text{Cut}$$

Note that not only do formulas of the form $!A$ in the antecedent of a sequent undergo weakening and contraction, but also they can be moved anywhere within the antecedent. The following rules are derivable using $\mathbf{!W}$, $\mathbf{!C}_1$, and $\mathbf{!C}_2$:

$$\frac{\Gamma, !A, \Delta, \Pi \vdash C}{\Gamma, \Delta, !A, \Pi \vdash C}\ \mathbf{!E}_1 \qquad \frac{\Gamma, \Delta, !A, \Pi \vdash C}{\Gamma, !A, \Delta, \Pi \vdash C}\ \mathbf{!E}_2$$

Formulas not of this restricted form do not undergo any of the usual structural rules, not even Exchange.

We will consider the *multiplicative Lambek calculus* (**MLC**), the calculus which has only the multiplicatives as its logical connectives; the *multiplicative-additive Lambek calculus* (**MALC**), which has the mulitplicatives and the additives but not the exponential; and the *multiplicative-exponential Lambek calculus* (**MELC**), which has the mulitplicatives and the exponential. The full Lambek calculus with all the connectives mentioned so far has the same recognizing power and complexity as the multiplicative-exponential fragment. What is known about the recognizing power and complexity of the three calculi is

summarized in Table 1. The complexity of **MLC** and the recognizing power of **MALC** have not been precisely characterized. In the following sections, we will review these results in turn, starting with **MELC**.

	recognizing power	complexity
MLC	$=$ CFL	$\in$ NP
MALC	$\supset$ finite intersections of CFLs	PSPACE-complete
MELC	$=$ r.e.	undecidable (r.e.-complete)

Table 1: Recognizing power and complexity of various fragments of the Lambek calculus.

3 Multiplicative-exponential Lambek calculus

Lincoln et al. (1992) show that **MELC** is undecidable by a reduction from semi-Thue systems. Their proof is given in the one-sided cyclic format for noncommutative linear logic, but the idea is even more transparent with the two-sided intuitionistic format adopted here. The proof provides an exact characterization of the complexity of **MELC**: it is complete for the r.e. sets (under computable many-one reductions). It also shows that **MELC** recognizes all r.e. languages.

The main idea is coding of **MLC** *theories* in **MELC** formulas. Since **MLC** provability in a theory, even for the fragment **MLC**($\otimes$) with $\otimes$ as the only connective, is quite easily shown to be undecidable, the coding can be used to show the undecidability of **MELC** (or **MELC**($\otimes, \multimap, !$)—$\multimap$ is necessary for the coding purpose).[2]

A K *theory* is just a finite subset of Seq_K. If T is a K theory, a sequent of K is provable in $K + T$ if there is a derivation of it which is just like a proof in K (possibly with applications of Cut) except that sequents in T may appear at some leaves.

The following lemma about **MLC** may be proved using standard cut elimination techniques.

Lemma 1. *Let T be an* **MLC** *theory. If an* **MLC** *sequent t is provable in* **MLC** $+ T$*, then there is a proof of t in* **MLC** $+ T$ *where every application of Cut involves an axiom in T as one of its premises.*

For any theory $T = \{t_1, \ldots, t_l\}$, its translation $[T]$ is defined to be the sequence

$$![t_1], \ldots, ![t_l],$$

where for any sequent $A_1, \ldots, A_n \vdash B$, $[A_1, \ldots, A_n \vdash B]$ is the formula

$$(A_1 \otimes \cdots \otimes A_n) \multimap B.$$

[2]We sometimes denote a calculus with a restricted set of connectives by displaying the connectives, as we do here.

Lemma 2. *Let T be an* **MLC** *theory, and let $\Pi \vdash D$ be an* **MLC** *sequent. Then the following are equivalent:*

(i) *$\Pi \vdash D$ is provable in* **MLC** $+ T$.

(ii) *$[T], \Pi \vdash D$ is provable in* **MELC**.

Proof. (i) $\Rightarrow$ (ii). By induction on the depth of proof, noting that if $A_1, \ldots, A_n \vdash B \in T$,

$$[T], A_1, \ldots, A_n \vdash B$$

is provable in **MELC**.

(ii) $\Rightarrow$ (i). Remove all occurrences of formulas of the form $!t$ from a proof of $[T], \Pi \vdash D$. All applications of rules for the mulitplicatives remain applications of the same rule. Applications of $!\mathbf{W}$, $!\mathbf{C}_1$, and $!\mathbf{C}_2$ become repetitions of the same sequent, so just delete one of the two occurrences. Applications of $(! \vdash)$ change as follows:

$$\frac{\begin{array}{c}\vdots\\ \Gamma, [A_1, \ldots, A_n \vdash B], \Delta \vdash C\end{array}}{\Gamma, ![A_1, \ldots, A_n \vdash B], \Delta \vdash C} \rightsquigarrow \frac{\begin{array}{c}\vdots\\ \Gamma', [A_1, \ldots, A_n \vdash B], \Delta' \vdash C\end{array}}{\Gamma', \Delta' \vdash C}$$

Then we can change all such parts simultaneously as follows:

$$\frac{\begin{array}{c}A_1, \ldots, A_n \vdash B\\ \vdots\\ \vdash [A_1, \ldots, A_n \vdash B]\end{array} \quad \begin{array}{c}\\ \\ \Gamma', [A_1, \ldots, A_n \vdash B], \Delta' \vdash C\end{array}}{\Gamma', \Delta' \vdash C}\ \text{Cut}$$

making the resulting figure a proof of $\Pi \vdash D$ in **MLC** $+ T$. ⊣

Note that in the above proof it is crucial to allow a sequent to have an empty antecedent.

We now describe a reduction from type 0 grammars of Chomsky to **MELC**$(\otimes, \multimap, !)$. A type 0 grammar G is a quadruple $\langle \Sigma, N, S, P \rangle$, where Σ is a finite alphabet, N is a finite set of *nonterminals* disjoint from Σ, S is a designated member of N, and P is a finite set of rewriting rules of the form

$$\alpha \to \beta,$$

where $\alpha \in (\Sigma \cup N)^+$ and $\beta \in (\Sigma \cup N)^*$. The 'rewrites' relation $\Rightarrow_G^*$ associated with G is defined by the following axioms and rule:

$$\frac{}{\alpha \Rightarrow^* \alpha} \text{ for } \alpha \in (\Sigma \cup N)^+ \qquad \frac{}{\alpha \Rightarrow^* \beta} \text{ for } \alpha \to \beta \in P \qquad \frac{\alpha \Rightarrow^* \beta \quad \gamma \Rightarrow^* \delta\, \alpha\, \sigma}{\gamma \Rightarrow^* \delta\, \beta\, \sigma}$$

$\alpha \Rightarrow_G^* \beta$ is defined to hold if $\alpha \Rightarrow^* \beta$ is derivable using the above axioms and rule. This definition of $\Rightarrow_G^*$ is diffrent from, but equivalent to, the standard definition. The *language generated by* G, denoted $\mathrm{L}(G)$, is defined to be $\{\, w \in \Sigma^* \mid S \Rightarrow_G^* w \,\}$.

Given a type 0 grammar $G = \langle \Sigma, N, S, P \rangle$, we define an **MLC** theory T_G as follows. Suppose $\Sigma = \{c_1, \ldots, c_m\}$ and let $N = \{D_1, \ldots, D_n\}$. We put $\Sigma \cup N = \{c_1, \ldots, c_m\} \cup \{D_1, \ldots, D_n\}$ into a one-one correspondence π with a set $\{p_1, \ldots, p_{m+n}\}$ of atomic formulas. For any expression of the form

$$X_1 \ldots X_i \Rightarrow^* Y_1 \ldots Y_j,$$

where $X_1, \ldots, X_i, Y_1, \ldots, Y_j \in \Sigma \cup N$ and $i \geq 1$, we associate the following sequent as its translation $\tau(X_1 \ldots X_i \Rightarrow^* Y_1 \ldots Y_j)$:

$$q_1, \ldots, q_j \vdash r_1 \otimes \cdots \otimes r_i,$$

where $q_1, \ldots, q_j$ and $r_1, \ldots, r_i$ are atomic formulas that correspond to $Y_1, \ldots, Y_j$ and $X_1, \ldots, X_i$, respectively, by π (note the flip-flop of the two sides). Let

$$T_G = \{\, \tau(\alpha \Rightarrow^* \beta) \mid \alpha \to \beta \in P \,\}.$$

We have the following:

Lemma 3. *For any $\alpha \in (\Sigma \cup N)^+$ and $\beta \in (\Sigma \cup N)^*$, $\alpha \Rightarrow_G^* \beta$ holds if and only if $\tau(\alpha \Rightarrow^* \beta)$ is provable in* **MLC** $+ T_G$.

The proof is straightforward, given our definition of $\Rightarrow_G^*$.

By Lemmas 2 and 3, we get

Lemma 4. *Let $G = \langle \Sigma, N, S, P \rangle$ be a type 0 grammar. For any $w \in \Sigma^*$, $w \in \mathrm{L}(G)$ if and only if*

$$[T_G], \Gamma \vdash q$$

is provable in **MELC**, *where $\Gamma \vdash q = \tau(S \Rightarrow^* w)$.*

Since the question '$w \in \mathrm{L}(G)$?' is undecidable, we get

Theorem 5 (Lincoln et al.). *Provability in* **MELC** *is undecidable.*

In fact, since type 0 grammars generate all r.e. sets, it follows that the set $\mathrm{Prov}_{\mathbf{MELC}}$ is complete for the r.e. sets.

Also, note that

$$[T_G], \Gamma \vdash D$$

is provable if and only if

$$\Gamma \vdash \bigotimes [T_G] \multimap D$$

is, where $\bigotimes(A_1, \ldots, A_n)$ is $A_1 \otimes \cdots \otimes A_n$. Then Lemma 4 implies

Theorem 6. **MELC** *recognizes all r.e. languages.*

The present reduction uses only the connectives $\otimes$, $\multimap$, and !, so the above results hold of **MELC**$(\otimes, \multimap, !)$. Kanovich (1993) shows that **MELC**$(\multimap, \circ\!\!-, !)$ is undecidable by a reduction from a two-counter machine. In fact, Buszkowski (1982) has shown that provability in **MLC**$(\multimap)$ theories is undecidable by a reduction from type 0 grammars. Since we can use $A_n \multimap (\dots \multimap (A_1 \multimap B) \dots)$ in place of $(A_1 \otimes \cdots \otimes A_n) \multimap B$ in Lemma 2 and Theorem 6, it follows that **MELC**$(\multimap, !)$ already has the same complexity and recognizing power as the full (multiplicative-additive-exponential) Lambek calculus.

As noted above, it is crucial to allow sequents with empty antecedent to capture provability in **MLC** theories by provability in **MELC**. To the best of my knowledge, there is no published proof that the **MELC** with Lambek's nonempty antecedent requirement has the same complexity and recognizing power as the **MELC** without the requirement.[3]

4 Multiplicative-additive Lambek calculus

Lincoln et al. (1992) show that multiplicative-additive linear logic is PSPACE-complete by reducing quantified Boolean formula satisfiability (QBF). Adapting their proof, we can show the same for multiplicative-additive Lambek calculus (**MALC**). (I arrived at this proof in ignorance of Kanovich 1994, where a similar proof is provided.) The key observation here is that satisfiability of closed quantified Boolean formulae whose quantifier-free matrix is in disjunctive normal form (DNF) is already PSPACE-complete. This enables us to do away with the Exchange rule which was necessary in Lincoln et al.'s encoding. Linear negation also turns out to be unnecessary.

Let $\{x_1, x_2, x_3, \dots\}$ be the set of propositional variables. Without loss of generality, we can assume that a closed quantified Boolean formula is of the form

$$Q_1 x_1 \dots Q_n x_n (D_1 \vee \cdots \vee D_m), \tag{1}$$

where for each i, Q_i is either $\forall$ or $\exists$ and D_i is a conjunction of literals from $\{x_1, \dots, x_n, \neg x_1, \dots, \neg x_n\}$ such that for every j, at most one of x_j and $\neg x_j$ occurs in D_i. Satisfiability of such formulas is PSPACE-complete. This follows from the well-known fact that satisfiability of quantified Boolean formulae with conjunctive normal form (CNF) matrix is PSPACE-complete and the fact that PSPACE = co-PSPACE.

We first show how the quantifer-free matrix $M = D_1 \vee \cdots \vee D_m$ of (1) can be encoded in **MALC**. Corresponding to the literals in M, the **MALC** encoding uses atomic formulas $x_1, \bar{x}_1, \dots, x_n, \bar{x}_n$. Each disjunct is translated as follows:

$$[\![D_i]\!] = y_1 \otimes \cdots \otimes y_n,$$

[3] In personal communication (1996), Max Kanovich informed me that **MELC** with Lambek's requirement is undecidable.

where

$$y_j = \begin{cases} x_j & \text{if } x_j \text{ occurs in } D_i, \\ \bar{x}_j & \text{if } \neg x_j \text{ occurs in } D_i, \\ (x_j \oplus \bar{x}_j) & \text{otherwise.} \end{cases}$$

Note that in the translation of D_i, the propositional variables appear in the sorted order. Then the translation of the DNF matrix is given by

$$[\![D_1 \vee \cdots \vee D_m]\!] = [\![D_1]\!] \oplus \cdots \oplus [\![D_m]\!].$$

If I is an assignment of truth values to propositional variables $x_1, \ldots, x_n$, then we let

$$[\![I]\!] = y_1, \ldots, y_n$$

where

$$y_j = \begin{cases} x_j & \text{if } I(x_j) = \text{true}, \\ \bar{x}_j & \text{if } I(x_j) = \text{false}. \end{cases}$$

The following lemma is straightforward.

Lemma 7. *Let M be a DNF formula in propositional variables $x_1, \ldots, x_n$, and let I be an assignment of truth values to $x_1, \ldots, x_n$. Then $I \models M$ if and only if $[\![I]\!] \vdash [\![M]\!]$ is provable in* **MALC**.

To translate quantified formulas, we introduce new atomic formulas $q_0, q_1, \ldots, q_n$. We translate each quantifier as follows:

$$[\![\forall x_i]\!] = q_{i-1} \multimap ((x_i \otimes q_i) \oplus (\bar{x}_i \otimes q_i)),$$
$$[\![\exists x_i]\!] = q_{i-1} \multimap ((x_i \otimes q_i) \mathbin{\&} (\bar{x}_i \otimes q_i)).$$

This is the 'lock-and-key' method for simulating quantifier dependencies introduced by Lincoln et al. (1992).

Lemma 8. *Let M be a DNF formula in propositional variables $x_1, \ldots, x_n$, and let I be an assignment of truth values to $x_1, \ldots, x_i$ $(0 \leq i \leq n)$. Then $I \models Q_{i+1}x_{i+1} \ldots Q_n x_n M$ if and only if*

$$[\![I]\!], q_i, [\![Q_{i+1}x_{i+1}]\!], \ldots, [\![Q_n x_n]\!] \vdash [\![M]\!] \otimes q_n$$

is provable in **MALC**.

As a special case, we have

Corollary 9. *Let M be a DNF formula in propositional variables $x_1, \ldots, x_n$. Then $\models Q_1 x_1 \ldots Q_n x_n M$ if and only if*

$$q_0, [\![Q_1 x_1]\!], \ldots, [\![Q_n x_n]\!] \vdash [\![M]\!] \otimes q_n$$

is provable in **MALC**.

This is a polynomial-time reduction from QBF satisfiability to provability in **MALC**. As in the case of mulitiplicative-additive linear logic, **MALC** is in PSPACE, so we have

Theorem 10. *Provability in* **MALC** *is PSPACE-complete.*

In contrast, the recognizing power of **MALC** has not been precisely characterized. There is a known lower bound, however, due to Kanazawa (1992).

Theorem 11. $\mathrm{Rec}_{\mathbf{MALC}(\multimap,\&)}$ *properly includes the class of finite intersections of context-free languages.*

Theorem 11 is a consequence of the following lemma and Lemma 15 below.

Lemma 12. $\mathrm{Rec}_{\mathbf{MALC}(\ldots,\&)}$ *(with any selection of connectives including* $\&$*) is closed under intersection.*

Proof. We give a more streamlined proof than Kanazawa 1992. Let $G_1 = \langle \mapsto_1, D_1 \rangle$ and $G_2 = \langle \mapsto_2, D_2 \rangle$ be $\mathbf{MALC}(\ldots,\&)$-grammars. Let $\mathbf{P}_1$ and $\mathbf{P}_2$ be the sets of atomic formulas used in G_1 and G_2, respectively. Without loss of generality, we can assume that $\mathbf{P}_1 \cap \mathbf{P}_2 = \emptyset$. Let $G = \langle \mapsto, D_1 \,\&\, D_2 \rangle$, where

$$\mapsto = \{\, \langle c, A \,\&\, B\rangle \mid c \mapsto_1 A, c \mapsto_2 B \,\}.$$

It suffices to prove $\mathrm{L}(G) = \mathrm{L}(G_1) \cap \mathrm{L}(G_2)$. This follows from the following claim. Below, formulas built exclusively from atomic formulas in $\mathbf{P}_i$ are called $\mathbf{P}_i$*-formulas*, for $i = 1, 2$.

Claim If $A_1, \ldots, A_n$ are $\mathbf{P}_1$-formulas and $B_1, \ldots, B_n$ are $\mathbf{P}_2$-formulas, then the following are equivalent:

(i) $A_1 \,\&\, B_1, \ldots, A_n \,\&\, B_n \vdash D_1 \,\&\, D_2$ is provable in **MALC**.

(ii) $A_1, \ldots, A_n \vdash D_1$ and $B_1, \ldots, B_n \vdash D_2$ are both provable in **MALC**.

Proof of the claim. (ii) $\Rightarrow$ (i). This direction is easy:

$$\dfrac{\dfrac{\dfrac{A_1, \ldots, A_n \vdash D_1}{A_1 \,\&\, B_1, \ldots, A_n \vdash D_1}}{\vdots}}{A_1 \,\&\, B_1, \ldots, A_n \,\&\, B_n \vdash D_1} \qquad \dfrac{\dfrac{\dfrac{B_1, \ldots, B_n \vdash D_2}{A_1 \,\&\, B_1, \ldots, B_n \vdash D_2}}{\vdots}}{A_1 \,\&\, B_1, \ldots, A_n \,\&\, B_n \vdash D_2}$$

$$\dfrac{A_1 \,\&\, B_1, \ldots, A_n \,\&\, B_n \vdash D_1 \quad A_1 \,\&\, B_1, \ldots, A_n \,\&\, B_n \vdash D_2}{A_1 \,\&\, B_1, \ldots, A_n \,\&\, B_n \vdash D_1 \,\&\, D_2}$$

(i) $\Rightarrow$ (ii). Assume (i). We show that $A_1, \ldots, A_n \vdash D_1$ is provable in **MALC**. By symmetry, the same holds for $B_1, \ldots, B_n \vdash D_2$.

First, note that $A_1 \,\&\, B_1, \ldots, A_n \,\&\, B_n \vdash D_1$ is provable:

$$\dfrac{A_1 \,\&\, B_1, \ldots, A_n \,\&\, B_n \vdash D_1 \,\&\, D_2 \quad \dfrac{D_1 \vdash D_1}{D_1 \,\&\, D_2 \vdash D_1}}{A_1 \,\&\, B_1, \ldots, A_n \,\&\, B_n \vdash D_1}\ \text{Cut}$$

Since Cut is eliminable in **MALC**, we get a cut-free proof $\mathcal{P}$ of $A_1 \,\&\, B_1, \ldots, A_n \,\&\, B_n \vdash D_1$. There are three possible ways in which a negative occurrence of $A_i \,\&\, B_i$ can originate in a cut-free proof in **MALC**:

(1) It originates in an axiom:

$$A_i \,\&\, B_i \vdash A_i \,\&\, B_i$$

(2) It originates in the conclusion of $(\& \vdash)$:

a.

$$\frac{\ldots, A_i, \ldots \vdash \ldots}{\ldots, A_i \,\&\, B_i, \ldots \vdash \ldots}$$

b.

$$\frac{\ldots, B_i, \ldots \vdash \ldots}{\ldots, A_i \,\&\, B_i, \ldots \vdash \ldots}$$

We will show that in $\mathcal{P}$, each $A_i \,\&\, B_i$ originates in the way indicated in (2a). If this is so, by skipping the applications of $(\& \vdash)$ that introduce $A_i \,\&\, B_i$ (for $i = 1, \ldots, n$), we can transform $\mathcal{P}$ into a proof of $A_1, \ldots, A_n \vdash D_1$.

Since there is no positive occurrence of $A_i \,\&\, B_i$ in $\mathcal{P}$, (1) is clearly excluded. We can show that (2b) is also impossible in $\mathcal{P}$ by two lemmas.

Lemma 13. *Let Γ and Δ be sequences consisting of $\mathbf{P}_1$-formulas, $\mathbf{P}_2$-formulas, and formulas of the form $A \,\&\, B$, where A is a $\mathbf{P}_1$-formula and B is a $\mathbf{P}_2$-formula. Let D be a $\mathbf{P}_1$-formula, and C be a $\mathbf{P}_2$-formula. Then $\Gamma, C, \Delta \vdash D$ is not provable in* **MALC**.

Proof. By induction on the depth of cut-free proof. ⊣

Lemma 14. *In $\mathcal{P}$, the succedent of every sequent is a $\mathbf{P}_1$-formula.*

Proof. Suppose there is a sequent in $\mathcal{P}$ with a $\mathbf{P}_2$-formula as its succedent. Since the succedent of the endsequent is a $\mathbf{P}_1$-formula, the lowest such sequent must be the left premise of an applicatoin of $(\multimap \vdash)$ or $(\circ\!\!- \vdash)$. Then the right premies violates Lemma 13. ⊣

This concludes the proof of the claim and Lemma 12. ⊣

Lemma 15. *There is no calculus K such that Rec_K equals the class of finite intersections of context-free languages.*

Proof. It suffices to point out that the class of finite intersections of context-free languages is not closed under strictly alphabetic morphisms (Păun 1980). Consider the following context-free languages.

$$\begin{aligned}
L_0 &= \{\, \mathtt{a}^i\mathtt{b}^i \mid i \geq 1 \,\} \\
L_1 &= L_0^* \\
L_2 &= \{\, \mathtt{b}^i\mathtt{a}^i \mid i \geq 1 \,\} \\
L_3 &= \bigcup_{n \geq 1} \mathtt{a}^n L_2^{n-1} \mathtt{b}^+
\end{aligned}$$

Then

$$L_1 \cap L_3 = \{\, (\mathtt{a}^n\mathtt{b}^n)^n \mid n \geq 1 \,\}.$$

Let h be a strictly alphabetic morphism such that $h(\mathtt{a}) = h(\mathtt{b}) = \mathtt{a}$. Then

$$h(L_1 \cap L_2) = \{\, \mathtt{a}^{2n^2} \mid n \geq 1 \,\},$$

which is not regular and hence cannot be a finite intersection of context-free languages, since all context-free subsets of $\mathtt{a}^*$ are regular. ⊣

Here are some open questions in view of Theorems 10 and 11:

- Does **MALC** recognize all context-sensitive languages?
- Does **MALC** recognize any PSPACE-complete languages? (Note that any finite intersection of context-free languages is in P.)
- Is **MALC**$(-\!\circ, \&)$ PSPACE-complete?

5 Multiplicative Lambek calculus

The recognizing power of the mulitplicative Lambek calculus **MLC** (or even the smaller fragment **MLC**$(-\!\circ)$) had been the most outstanding open question about the mathematics of the Lambek calculus until 1992, when Mati Pentus announced a proof of the so-called 'Chomsky conjecture' (Pentus 1993):

Theorem 16 (Pentus). $\mathrm{Rec}_{\mathbf{MLC}}$ *is precisely the class of all context-free languages.*

Pentus's proof shows that given any Lambek grammar, one can effectively find an equivalent context-free grammar. Given $\Gamma \vdash C$, let $\mathbf{A}$ be $\{\, A \mid A$ is not longer than the longest formula in $\Gamma \vdash C$ and contains only atomic formulas in $\Gamma \vdash C \,\}$. The main step of the proof is to show that if $\Gamma \vdash C$ is provable in **MLC**, then $\Gamma \vdash C$ is provable from the following set by Cut alone:

$$\{\, A_1 \vdash A_2 \in \mathrm{Prov}_{\mathbf{MLC}} \mid A_1, A_2 \in \mathbf{A} \,\} \cup$$
$$\{\, A_1, A_2 \vdash A_3 \in \mathrm{Prov}_{\mathbf{MLC}} \mid A_1, A_2, A_3 \in \mathbf{A} \,\}$$

It is not known whether the construction of this set can be carried out in polynomial time, so it does not follow that $\mathrm{Prov}_{\mathbf{MLC}}$ is in P, although once the set is constructed, provability by Cut alone can be checked by reduction to universal context-free recognition, which is in P. $\mathrm{Prov}_{\mathbf{MLC}}$ is clearly in NP, but nobody has shown that it is in P or is NP-hard.

Pentus's theorem shows that for any finite set $\mathbf{A}$ of **MLC** formulas and any **MLC** formula B, the set

$$\mathrm{Prov}^{\mathbf{A},B}_{\mathbf{MLC}} = \mathrm{Prov}_{\mathbf{MLC}} \cap \{\, \Gamma \vdash B \mid \Gamma \in \mathbf{A}^* \,\}$$

is a context-free language. Could $\mathrm{Prov}_{\mathbf{MLC}}$ be a context-free language? If so, this would of course imply that $\mathrm{Prov}_{\mathbf{MLC}}$ is in P. It is easy to see, however, that

the answer is no, even over a finite set of atomic formulas.[4] A more interesting question to ask would be whether there is an easily computable function f that reduces $\mathrm{Prov}_{\mathbf{MLC}}$ to a context-free set.

It is also worth mentioning that another result of Pentus's (1994) implies that the symmetric transitive closure of the '$\vdash$ relation' between formulas

$$\{\, \langle A, B\rangle \mid A \vdash B \in \mathrm{Prov}_{\mathbf{MLC}} \,\}$$

is in P. In fact, there is an easy transformation $[\![\cdot]\!]$ (Pentus's 'free group' interpretation) that makes the set

$$\{\, [\![A \vdash B]\!] \mid A \vdash B \in \mathrm{Prov}_{\mathbf{MLC}} \,\}$$

context-free, and this constitutes a reduction of the relation in question.

Another interesting consequence of Pentus's (1993) theorem is that the following question is decidable.[5]

> **STRONG DEDUCIBILITY IN MLC.**
> INSTANCE: a finite set $\{A_1, \ldots, A_n\}$ of **MLC** formulas and an **MLC** formula B.
> QUESTION: Is there a finite sequence $\Gamma \in \{A_1, \ldots, A_m\}^*$ such that $\Gamma \vdash B$ is provable in **MLC**?

This is decidable since non-emptyness of context-free languages is decidable.

One might find the decidability of strong deducibility in **MLC** puzzling since it looks very similar to the question of whether a sequent of the form

$$!A_1, \ldots, !A_n \vdash B,$$

where $A_1, \ldots, A_n, B$ are **MLC** formulas, is provable in **MELC**; the latter question is undecidable by the proof of Theorem 5. In fact, the linear logic counterparts of the two questions are equivalent (see Ono 1998). This is not so in the Lambek calculus. The following simple example shows that the two questions are not equivalent.

$$\dfrac{\dfrac{\dfrac{\dfrac{p_1, p_2, p_3 \vdash p_1 \otimes p_2 \otimes p_3}{p_1, !p_2, p_3 \vdash p_1 \otimes p_2 \otimes p_3}}{\vdots}}{p_1, p_3, !p_2 \vdash p_1 \otimes p_2 \otimes p_3}}{p_1 \otimes p_3, !p_2 \vdash p_1 \otimes p_2 \otimes p_3}}{!(p_1 \otimes p_3), !p_2 \vdash p_1 \otimes p_2 \otimes p_3}$$

As the above proof shows, the sequent $!(p_1 \otimes p_3), !p_2 \vdash p_1 \otimes p_2 \otimes p_3$ is provable, but there is no sequence $\Gamma \in \{p_1 \otimes p_3, p_2\}^*$ such that $\Gamma \vdash p_1 \otimes p_2 \otimes p_3$ is provable.

[4] A sequent of the form:

$$p_0 \multimap (p_1 \multimap (\ldots \multimap p_m) \ldots) \vdash q_0 \multimap (q_1 \multimap (\ldots \multimap q_n) \ldots)$$

is provable if and only if $m = n$ and $p_i = q_i$ for $i = 0, \ldots, m$.

[5] The term *strong reducibility* is from Ono 1998. This was called $*$-derivability by van Benthem (1991).

Note also that the above result about the recognizing power of **MALC** (Theorem 11) implies that **MALC** has an undecidable strong deducibility problem, in contrast to **MLC**. This follows from the fact that the question of whether given two context-free languages have a non-empty intersection is undecidable. Here we see that a language-theoretic characterization of the recognizing power of a logic can help to settle more standard questions about the logic.

References

van Benthem, J. 1991. *Language in Action*. Amsterdam: North-Holland. (Second edition 1995, MIT Press.)

Buszkowski, W. 1982. Some decision problems in the theory of syntactic categories. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik* **28**, 539–548.

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

Kanovich, M. I. 1993. The expressive power of modalized purely implicational calculi. CSLI Report. Center for the Study of Language and Information, Stanford University.

Kanovich, M. I. 1994. Horn fragments of non-commutaive logics with additives are PSPACE-complete. In *Proceedings 1994 Annual Conference of the European Association for Computer Science Logic*. Kazimierz, Poland.

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*, pp. 161–178. American Mathematical Society.

Lincoln, P., J. Mitchell, A. Scedrov, and N. Shankar. 1992. Decision problems for propositional linear logic. *Annals of Pure and Applied Logic* **56**, 239–311.

Ono, H. 1998. Decidability and finite model property of substructural logics. In J. Ginzburg, Z. Khasidashvili, C. Vogel, J.J. Lévy, and E. Vallduví, eds., *The Tbilisi Symposium on Logic, Language and Computation: Selected Papers*. Stanford, Calif.: CSLI Publications.

Păun, G. 1980. A note on the intersection of context-free languages. *Fundamenta Informticae* **3**, 135–139.

Pentus, M. 1993. Lambek grammars are context free. In *Proceedings of the Eighth Annual IEEE Symposium on Logic in Computer Science*, pp. 429–433.

Pentus, M. 1994. Equivalent types in Lambek calculus and linear logic. *Journal of Logic, Language and Information* **3**, 121–140.

Pentus, M. 1997. Product-free Lambek calculus and context-free grammars. *Journal of Symbolic Logic* **62**, 648–660.