On some Extensions of Syntactic Concept Lattices: Completeness and Finiteness Results

Christian Wurm
cwurm@phil.hhu.de

Universität Düsseldorf

Abstract. We provide some additional completeness results for the full Lambek calculus and syntactic concept lattices, where the underlying structure is extended to tuples of arbitrary finite and infinite size. Whereas this answers an open question for finite tuples, infinite tuples have not been considered yet. Nonetheless, they have a number of interesting properties which we establish in this paper, such as a particular class of languages which results in a finite lattice.

1 Introduction

Syntactic concept lattices arise from the distributional structure of languages. Their main advantage is that they can be constructed on distributional relations which are weaker than strict equivalence. [3] has shown how these lattices can be enriched with a monoid structure to form residuated lattices. [19] has shown that the resulting class of syntactic concept lattices for arbitrary languages forms a complete class of models for the logics $\mathbf{FL}_\perp$, i.e. the full Lambek calculus, and its reducts $\mathbf{FL}, L1$, for which it is a conservative extension.

In this paper, we will consider syntactic concept lattices over extended monoids: these will no longer consist of (sets of) strings, but rather of (sets of) tuples of strings, first of arbitrary finite, then of infinite size. The monoid operation has to be modified accordingly, of course. We show that the completeness results can be extended to this case for $\mathbf{FL}_\perp$ and its reducts; our proof will be constructed on top of the completeness results in [19] by means of isomorphic embeddings.

Finite tuples have been considered in formal language theory in a huge number of different contexts; the most relevant for us are [5],[15]. The use of infinite tuples has not been considered yet (to our knowledge). We show that it comes with some interesting gain in expressive power, while still being well-behaved; we also establish the largest class of language which results in a finite lattice over infinite tuples.

2 Residuated Syntactic Concept Lattices and Extensions

2.1 Equivalences on Strings and Tuples

Syntactic concept lattices originally arose in the structuralist approach to syntax, back when syntacticians tried to capture syntactic structures purely in terms of distributions of strings[1] (see, e.g. [10]). An obvious way to do so is by partitioning strings/substrings into *equivalence classes*: we say that two strings w, v are equivalent in a language $L \subseteq \Sigma^*$, in symbols

(1) $\quad w \sim^1_L v$, iff for all $x, y \in \Sigma^*$, $xwy \in L \Leftrightarrow xvy \in L$.

This can be extended to tuples of strings of arbitrary size:

(2) $\quad (w_1, v_1) \sim^2_L (w_2, v_2)$, iff for all $x, y, z \in \Sigma^*$, $xw_1yv_1z \in L \Leftrightarrow xw_2yv_2z$, etc.

The problem with equivalence classes is that they are too restrictive for many purposes: assume we want to induce our grammar on the basis of a given dataset; then it is quite improbable that we get the equivalence classes we would usually desire. And even if we have an unlimited supply of examples, it seems unrealistic to describe our grammar on the basis of equivalence classes only: there might be constructions, collocations, idioms which ruin equivalences which we would intuitively consider to be adequate. Another drawback of equivalence classes is that for context-free languages, there is no general way to relate them to the non-terminals of some grammar generating the language, whereas for syntactic concepts, there are some interesting connections (see [6]).

Syntactic concepts provide a somewhat less rigid notion of equivalence, which can be conceived of as equivalence restricted to a given set of contexts. This at least partly overcomes the difficulties we have mentioned here.

2.2 Syntactic Concepts and Polar Maps

For a general introduction to lattices, see [7]; for background on residuated lattices, see [9]. Syntactic concept lattices form a particular case of what is well-known as formal concept lattice (or formal concept analysis) in computer science. In linguistics, they have been introduced in [18]. They were brought back to attention and enriched with residuation in [3], [4], as they turn out to be useful representations for language learning.

Given a language $L \subseteq \Sigma^*$, $n \in \mathbb{N}$, we define two maps: a map $\triangleright : \wp((\Sigma^*)^n) \to \wp((\Sigma^*)^{n+1})$, and $\triangleleft : \wp((\Sigma^*)^{n+1}) \to \wp((\Sigma^*)^n)$, which are defined as follows:

(3) $\quad M^\triangleright := \{(x_1, ..., x_{n+1} : \forall (w_1, ..., w_n) \in M, x_1w_1...w_nx_{n+1} \in L\}$;

and dually

(4) $\quad C^\triangleleft := \{(w_1, ..., w_n) : \forall (x_1, ..., x_{n+1}) \in C, x_1w_1...w_nx_{n+1} \in L\}$.

That is, a set of tuples of strings is mapped to the set of tuples of contexts in which all of its elements can occur. The dual function maps a set of contexts to the set of strings, which can occur in all of them. Usually, the case where $n = 1$ has been in the focus, as in [3],[19]. The more general cases have been considered in one form or other by [5],[15]). Obviously, $\triangleleft$ and $\triangleright$ are only defined with respect

[1] Or words, respectively, depending on whether we think of our language as a set of words or a set of strings of words; we will choose the former option.

to a given language L and a given tuple size, otherwise they are meaningless. As long as it is clear of which language (if any particular language) and tuple size (if any particular) we are speaking, we will omit however any reference to it, to keep notation perspicuous. For a set of contexts C, $C^{\triangleleft}$ can be thought of as an equivalence class with respect to the contexts in C; but there might be elements in $C^{\triangleleft}$ which can occur in a context $(v, w) \notin C$ (and conversely). There is one more extension we will consider which is not entirely trivial, namely the one from tuples of *arbitrary* size to tuples of *infinite* size.

(5) for $M \subseteq (\Sigma^*)^\omega$, $M^{\triangleright} := \{(x_1, x_2, ...) : \forall (w_1, w_2, ...) \in M, x_1w_1x_2w_2... \in L\}$.

One might consider this meaningless, as L consists of finite words, M of infinite tuples. But this is unjustified: it only entails that for any infinite tuple $\overline{w} \in M$ or $\overline{w} \in M^{\triangleright}$, in order to be "meaningful", all but finitely many components must be ϵ. So for each "meaningful" $(w_1, w_2, ...)$, there is a $k \in \mathbb{N}$ such that for all $j \geq k$, $w_j = \epsilon$. We gladly accept this restriction and remain with tuples where *almost all* components are empty. This is still a proper generalization of any tuple size n, because there is no finite upper bound for non-empty components in sets of tuples.

We define $\cdot$ on (finite or infinite) tuples by componentwise concatenation, that is, $(w_1, w_2, ...) \cdot (v_1, v_2, ...) = (w_1v_1, w_2v_2, ...)$. This choice is not unquestionable: some authors seem to prefer concatenation of the type $\oplus$, where $(w, v) \oplus (x, y) = (wx, yv)$. In the context of multiple context-free grammars this is referred to as *well-nestedness* and has attracted great interest (see e.g. [12]). The problem with this type of concatenation is that it is not easily extended beyond tuples of size two. What is interesting in this context is that we can use the ω-tuples to *simulate* $\oplus$-style concatenation with $\cdot$-style concatenation. To briefly sketch what this means, we define the forgetful map fo by $\mathsf{fo}(w_1, w_2, ...) = w_1w_2...$, for arbitrary finite/infinite tuple size. We can now for all sequences of tuples $(x_1, y_1), ..., (x_i, y_i)$ devise ω-tuples $\overline{w}_1, ..., \overline{w}_i$ such that for all $1 \leq j, j' \leq i$, we have

$$\mathsf{fo}((x_j, y_j) \oplus ... \oplus (x_{j'}, y_{j'})) = \mathsf{fo}(\overline{w}_j \cdot ... \cdot \overline{w}_{j'})$$

This is not generally possible with any finite tuple size, and this is what makes infinite tuples interesting for us. Note that we can now also simplify things for ω-tuples, as we have $\overline{v} \in \{\overline{w}\}^{\triangleright}$ iff $\mathsf{fo}(\overline{v} \cdot \overline{w}) \in L$.

Regardless of the underlying objects, the two compositions of the maps, $\triangleleft\triangleright$ and $\triangleright\triangleleft$, are closure operators, that is:

1. $M \subseteq M^{\triangleright\triangleleft}$,
2. $M^{\triangleright\triangleleft} = M^{\triangleright\triangleleft\triangleright\triangleleft}$,
3. $M \subseteq N \Rightarrow M^{\triangleright\triangleleft} \subseteq N^{\triangleright\triangleleft}$,

for $M, N \subseteq \Sigma^*$. The same holds for contexts and $\triangleleft\triangleright$. A set M is **closed**, if $M^{\triangleright\triangleleft} = M$ etc. The closure operator $\triangleright\triangleleft$ gives rise to a lattice $\langle \mathcal{B}^n_L, \leq \rangle$, where the elements of $\mathcal{B}^n_L$ are the sets $M \subseteq (\Sigma^*)^n$ such that $M = M^{\triangleright\triangleleft}$, and $\leq$ is interpreted as $\subseteq$. The same can be done with the set of closed contexts. Given these two lattices, $\triangleright$ and $\triangleleft$ establish a Galois connection between the two:

1. $M \leq N \Leftrightarrow M^{\triangleleft} \geq N^{\triangleleft}$, and
2. $C \leq D \Leftrightarrow C^{\triangleright} \geq D^{\triangleright}$.

A syntactic concept A is usually defined to be an ordered pair, consisting of a closed set of strings, and a closed set of contexts, so it has the form $\langle S, C\rangle$, such that $S^{\triangleright} = C$ and $C^{\triangleleft} = S$. For our purposes, we only need to consider the left component, so we suppress the contexts and only consider the stringsets of the form $M^{\triangleright\triangleleft}$. For all operations we define below, it can be easily seen that the resulting structures are isomorphic. So when we refer to a concept, we only mean a $[-]^{\triangleright\triangleleft}$ closed set of strings, the concept in the classical sense being easily reconstructible.

Definition 1 *The set of concepts of a language L forms a lattice denoted by $SCL_n(L) := \langle \mathcal{B}^n_L, \wedge, \vee, \top, \bot\rangle$, where $\top = (\Sigma^*)^n$, $\bot = \emptyset^{\triangleright\triangleleft}$, and for $M, N \in \mathcal{B}^n_L$, $M \wedge N = M \cap N$, $M \vee N = (M \cup N)^{\triangleright\triangleleft}$.*

It is easy to see that this defines an order in the usual fashion which coincides with $\subseteq$ on closed sets of strings. It is easy to verify that this forms a complete lattice, as infinite joins are defined by (closure of) infinite unions, infinite meets by infinite intersections. Note also that for any set of (tuples of) strings S and contexts C, $S^{\triangleright} = S^{\triangleright\triangleleft\triangleright}$ and $C^{\triangleleft} = C^{\triangleleft\triangleright\triangleleft}$. $SCL_\omega(L)$ denotes the according structure with infinite tuple size. To see that things are properly different in the infinite case, we present the following result:

Lemma 2 *1. For any $n \in \mathbb{N}$, $SCL_n(L)$ is finite iff $L \in Reg$.*
2. There are $L \in Reg$ such that $SCL_\omega(L)$ is infinite.

Let $[\Sigma]^*_{\sim^1_L}$ denote the set of $\sim^1_L$-congruence classes over Σ^*.

Proof. 1. $SCL_1(L)$ is finite iff $[\Sigma]^*_{\sim^1_L}$ is finite iff L is regular (both are well-known). Moreover, $|[\Sigma^*]_{\sim^{n+1}_L}| \leq |[\Sigma^*]_{\sim^n_L}| \cdot |[\Sigma^*]_{\sim^1_L}|$, as $\sim^1_L$-equivalent strings are equivalent in all contexts. From this the claim easily follows.

2. Take the language $L = a^*b^*$, and all tuples of the form $(a, \overbrace{\epsilon, ..., \epsilon}^{n\ times}, a, \epsilon, \epsilon, ...)$ for $n \in \mathbb{N}$. For every $m, m' \in \mathbb{N}$, $m < m'$, we take the tuple $(\overbrace{a, ..., a}^{m+1\ times}, \overbrace{b, ..., b}^{m+3\ times}, \epsilon, \epsilon, ...)$; it is easy to see that $\mathsf{fo}((\overbrace{a, ..., a}^{m+1\ times}, \overbrace{b, ..., b}^{m+3\ times}, \epsilon, \epsilon, ...) \cdot (a, \overbrace{\epsilon, ..., \epsilon}^{m\ times}, a, \epsilon, \epsilon, ...)) \in L$, whereas if we substitute m with m', the result is not in L.

Consequently, there are infinitely concepts, namely for each $n \in \mathbb{N}$ at least one which contains $(a, \overbrace{\epsilon, ..., \epsilon}^{n\ times}, a, \epsilon, \epsilon, ...)$ but no $(a, \overbrace{\epsilon, ..., \epsilon}^{n'\ times}, a, \epsilon, \epsilon, ...)$ for $n' > n$. $\dashv$

This raises the question: what is the class C of languages such that $L \in C$ if and only if $SCL_\omega(L)$ is finite? We will give a concise characterization of this class later on.

2.3 Monoid Structure and Residuation

As we have seen, the set of concepts of a language forms a lattice. In addition, we can also give it the structure of a monoid: for concepts M, N, we define $M \circ N := (M \cdot N)^{\rhd\lhd}$, where $M \cdot N = \{\overline{w} \cdot \overline{v} : \overline{w} \in M, \overline{v} \in N\}$. We often write MN for $M \cdot N$. '$\circ$' is associative on concepts: For $M, N, O \in \mathcal{B}_n^L$, $M \circ (N \circ O) = (M \circ N) \circ O$. This follows from the fact that $[-]^{\rhd\lhd}$ is a *nucleus*, that is, it is a closure operator and in addition it satisfies $S^{\rhd\lhd}T^{\rhd\lhd} \subseteq (ST)^{\rhd\lhd}$, and the associativity of $\cdot$-concatenation (no matter on which tuple size).

Furthermore, it is easy to see that the neutral element of '$\circ$' is $\{\epsilon\}^{\rhd\lhd}$. The monoid operation respects the partial order of the lattice, that is, for $X, Y, Z, W \in \mathcal{B}_n^L$, if $X \leq Y$, then $W \circ X \circ Z \leq W \circ Y \circ Z$. A stronger property is the following: $\circ$ distributes over infinite joins, that is, we have

$$\bigvee_{Z \in \mathbf{Z}} X \circ Z \circ Y = X \circ \bigvee \mathbf{Z} \circ Y$$

$\leq$ follows algebraically ($\circ$ respects the order), and $\geq$ follows from the fact that $[-]^{\rhd\lhd}$ is a nucleus.

We enrich this with residuals, using the following definition:

Definition 3 *Let X, Y be concepts. We define the right residual $X/Y := \bigvee\{Z : Z \circ Y \leq X\}$, the left residual $Y\backslash X := \bigvee\{Z : Y \circ Z \leq X\}$.*

Note that this is an entirely abstract definition which does not make reference to any underlying structure. That it works is ensured by the following lemma.

Lemma 4 *Let L be a complete lattice with a monoid operation distributing over infinite joins. Then for $X, Y, Z \in L$, residuals defined as above, we have $Y \leq X\backslash Z$ iff $X \circ Y \leq Z$ iff $X \leq Z/Y$.*

Proof. We only prove the first bi-implication.

If: assume $X \circ Y \leq Z$. Then $Y \in \{W : X \circ W \leq Z\}$; so $Y \leq \bigvee\{W : X \circ W \leq Z\} = X\backslash Z$.

Only if: we have $X \circ X\backslash Z = X \circ \bigvee\{W : X \circ W \leq Z\} = \bigvee\{X \circ W : X \circ W \leq Z\} \leq Z$; as $Y \leq X\backslash Z$, we have $X \circ Y \leq X \circ X\backslash Z \leq Z$. $\dashv$

So every complete lattice with a monoid operation based on a nucleus can be extended to a residuated lattice.

Definition 5 *The **syntactic concept lattice** of a language L is defined as $SCL_n(L) := \langle \mathcal{B}_L^n, \wedge, \vee, \top, \bot, 1, \circ, /, \backslash \rangle$, where $\mathcal{B}_L^n, \wedge, \vee, \top, \bot$ are defined as in definition 1, $1 = \{\epsilon\}^{\rhd\lhd}$, and $\circ, /, \backslash$ are as defined above.*

Note that we somewhat overloaded the notation of $SCL_n(L)$; in the sequel we will however thereby always refer to definition 5. Moreover, we will denote by SCL the class of all lattices of the form $SCL_1(L)$ for some language L, without any further requirement regarding L; same for SCL_n for $n \in \mathbb{N} \cup \{\omega\}$.

3 Lambek Calculus and Extensions

3.1 The Logics L, $L1$, FL and FL$_\perp$

The Lambek calculus L was introduced in [13]. $L1$ is a proper extension of L, and $\mathbf{FL}, \mathbf{FL}_\perp$ are each conservative extensions of $L1$ and the preceding one. Let Pr be a set, the set of **primitive types**, and C be a set of **constructors**, which is, depending on the logics we use, $C_L := \{/, \backslash, \bullet\}$, or $C_{\mathbf{FL}} := \{/, \backslash, \bullet, \vee, \wedge\}$. By $Tp_C(Pr)$ we denote the set of types over Pr, which is defined as the smallest set, such that $Pr \subseteq Tp_C(Pr)$, and if $\alpha, \beta \in Tp_C(Pr)$, $\star \in C$, then $\alpha \star \beta \in Tp_C(Pr)$.

If there is no danger of confusion regarding the primitive types and constructors, we also simply write Tp for $Tp_C(Pr)$. We now present the inference rules corresponding to these constructors. We call an inference of the form $\Gamma \vdash \alpha$ a **sequent**, for $\Gamma \in Tp^*$, $\alpha \in Tp$, where by Tp^* we denote the set of all (possibly empty) *sequences* over Tp, which are concatenated by ',' (keep in mind the difference between *sequents*, which have the form $\Gamma \vdash \alpha$, and *sequences* like Γ, which are in Tp^*).

With few exceptions, rules of inference in our logics are not given in the form of sequents $\Gamma \vdash \alpha$, but rather as rules to derive new sequents from given ones. In general, uppercase Greek letters range as variables over sequences of types. In the inference rules for L, premises of $' \vdash '$ (that is, left hand sides of sequents) must be non-empty; in $L1$ they can be empty as well; everything else is equal. In **FL** and $\mathbf{FL}_\perp$ we also allow for empty sequents. Lowercase Greek letters range over single types. Below, we present the standard rules of the Lambek calculus L / $L1$.

$$(ax) \qquad \alpha \vdash \alpha \qquad\qquad (cut) \qquad \frac{\Delta, \beta, \Theta \vdash \alpha \quad \Gamma \vdash \beta}{\Delta, \Gamma, \Theta \vdash \alpha}$$

$$(\mathbf{I} - /) \qquad \frac{\Gamma, \alpha \vdash \beta}{\Gamma \vdash \beta/\alpha} \qquad\qquad (\mathbf{I} - \backslash) \qquad \frac{\alpha, \Gamma \vdash \beta}{\Gamma \vdash \alpha\backslash\beta}$$

$$(/ - \mathbf{I}) \qquad \frac{\Delta, \beta, \Theta \vdash \gamma \quad \Gamma \vdash \alpha}{\Delta, \beta/\alpha, \Gamma, \Theta \vdash \gamma} \qquad\qquad (\backslash - \mathbf{I}) \qquad \frac{\Delta, \beta, \Theta \vdash \gamma \quad \Gamma \vdash \alpha}{\Delta, \Gamma, \alpha\backslash\beta, \Theta \vdash \gamma}$$

$$(\bullet - \mathbf{I}) \qquad \frac{\Delta, \alpha, \beta, \Gamma \vdash \gamma}{\Delta, \alpha \bullet \beta, \Gamma \vdash \gamma} \qquad\qquad (\mathbf{I} - \bullet) \qquad \frac{\Delta \vdash \alpha \quad \Gamma \vdash \beta}{\Delta, \Gamma \vdash \alpha \bullet \beta}$$

These are the standard rules of L / $L1$ (roughly as in [13]). We have rules to introduce either slash and '$\bullet$' both on the right hand side of $\vdash$ and on the left hand side of $\vdash$. We will now add two additional connectives, which are well-known from structural logics, namely $\vee$ and $\wedge$. These are not present in $L/L1$, have however been considered as extensions as early as in [14], and have been subsequently studied by [11].

$$(\wedge - \mathbf{I}\ 1)\quad \frac{\Gamma,\alpha,\Delta\vdash\gamma}{\Gamma,\alpha\wedge\beta,\Delta\vdash\gamma}\qquad(\wedge - \mathbf{I}\ 2)\quad \frac{\Gamma,\beta,\Delta\vdash\gamma}{\Gamma,\alpha\wedge\beta,\Delta\vdash\gamma}$$

$$(\mathbf{I} - \wedge)\quad \frac{\Gamma\vdash\alpha\quad \Gamma\vdash\beta}{\Gamma\vdash\alpha\wedge\beta}$$

$$(\vee - \mathbf{I})\quad \frac{\Gamma,\alpha,\Delta\vdash\gamma\quad \Gamma,\beta,\Delta\vdash\gamma}{\Gamma,\alpha\vee\beta,\Delta\vdash\gamma}$$

$$(\mathbf{I} - \vee\ 1)\quad \frac{\Gamma\vdash\alpha}{\Gamma\vdash\alpha\vee\beta}\qquad(\mathbf{I} - \vee\ 2)\quad \frac{\Gamma\vdash\beta}{\Gamma\vdash\alpha\vee\beta}$$

$$(1 - \mathbf{I})\quad \frac{\Gamma,\Delta\vdash\alpha}{\Gamma,1,\Delta\vdash\alpha}\qquad(\mathbf{I} - 1)\quad \vdash 1$$

This gives us the logic **FL**. Note that this slightly deviates from standard terminology, because usually, **FL** has an additional constant 0 (not to be confused with $\bot$!). In our formulation, 0 and 1 coincide. In order to have logical counterparts for the bounded lattice elements $\top$ and $\bot$, we introduce two logical constants, which are denoted by the same symbol.[2]

$$(\bot - \mathbf{I})\quad \Gamma,\bot\ \Delta\vdash\alpha\qquad(\mathbf{I} - \top)\quad \Gamma\vdash\top$$

This gives us the calculus $\mathbf{FL}_\bot$. From a logical point of view, all these extensions of L are quite well-behaved: they are conservative, and also allow us to preserve the important result of [13], namely admissibility of the cut-rule.

We say that a sequent $\Gamma\vdash\alpha$ is derivable in a calculus, if it can be derived by the axiom and the rules of inference; we then write $\Vdash_L \Gamma\vdash\alpha$, $\Vdash_{L1} \Gamma\vdash\alpha$, $\Vdash_{\mathbf{FL}} \Gamma\vdash\alpha$ etc., depending on which calculus we use.

3.2 Interpretations of $L1$, FL and $\mathbf{FL}_\bot$

The standard model for $L1$ is the class of residuated monoids. These are structures $(M,\cdot,1,\backslash,/,\leq)$, where $(M,\cdot,1)$ is a monoid, $(M,\leq)$ is a partial order, and $\cdot,/,\backslash$ satisfy the law of residuation: for $m,n,o\in M$,

$$(6)\qquad m\leq o/n \Leftrightarrow m\cdot n\leq o\Leftrightarrow n\leq m\backslash o.$$

Note that this implies that $\cdot$ respects the order $\leq$. The standard model for **FL** is the class of residuated lattices, and for $\mathbf{FL}_\bot$, the class of bounded residuated

[2] Whereas L and $L1$ are equally powerful in the sense of languages which are recognizable, [11] shows that **FL** is considerably more powerful than L: whereas L only recognizes context-free languages by the classical result of [17], **FL** can recognize any finite intersection of context-free languages. We only briefly mention this, because we have no space to make precise what it means for a calculus to recognize a class of languages.

lattices. A residuated lattice is an algebraic structure $\langle M, \cdot, \vee, \wedge, \backslash, /, 1\rangle$, where in addition to the previous requirements, $(M, \vee, \wedge)$ is a lattice; the lattice order $\leq$ need not be stated, as it can be induced by $\vee$ or $\wedge$: for $a, b \in M$, $a \leq b$ is a shorthand for $a \vee b = b$. A bounded residuated lattice is a structure $\langle M, \cdot, \vee, \wedge, \backslash, /, 1, \top, \bot\rangle$, where $\langle M, \cdot, \vee, \wedge, \backslash, /, 1\rangle$ is a residuated lattice, $\top$ is the maximal element of the lattice order $\leq$ and $\bot$ is its minimal element.

For a general introduction see [9]. We will give definitions only once for each operator; we can do so because each definition for a given connector is valid for all classes in which it is present.

We call the class of residuated monoids RM, the class of residuated lattices RL; the class of bounded residuated lattices $RL_\bot$. We now give a semantics for the calculi above. We start with an interpretation $\sigma : Pr \to M$ which interprets elements in Pr in elements of the lattice, and extend σ to $\overline{\sigma}$ by defining it inductively over our type constructors, which is for now the set $C := \{/, \backslash, \bullet, \vee, \wedge\}$. For $\alpha, \beta \in Tp_C(Pr)$,

1. $\overline{\sigma}(\alpha) = \sigma(\alpha) \in M$, if $\alpha \in Pr$
2. $\overline{\sigma}(\top) = \top$

2' $\overline{\sigma}(\top)$ is an arbitrary $m \in M$ such for all $\alpha \in Tp_C(Pr)$, $\overline{\sigma}(\alpha) \leq m$.

3. $\overline{\sigma}(\bot) = \bot$
4. $\overline{\sigma}(1) = 1$
5. $\overline{\sigma}(\alpha \bullet \beta) := \overline{\sigma}(\alpha) \cdot \overline{\sigma}(\beta)$
6. $\overline{\sigma}(\alpha/\beta) := \overline{\sigma}(\alpha)/\overline{\sigma}(\beta)$
7. $\overline{\sigma}(\alpha\backslash\beta) := \overline{\sigma}(\alpha)\backslash\overline{\sigma}(\beta)$
8. $\overline{\sigma}(\alpha \vee \beta) := \overline{\sigma}(\alpha) \vee \overline{\sigma}(\beta)$
9. $\overline{\sigma}(\alpha \wedge \beta) := \overline{\sigma}(\alpha) \wedge \overline{\sigma}(\beta)$

Note that the constructors on the left-hand side and on the right-hand side of the definition look identical (with the exception of $\bullet$ and $\cdot$), but they are not: on the left-hand side, they are type constructors, on the right hand side, they are operators of a residuated lattice. The same holds for the constants $\top, \bot, 1$. Note that there are two alternative interpretations for $\top$: one which interprets it as the upper bound for the lattice, which is the standard interpretation, and one which just interprets it as an arbitrary element. The latter will be called the **non-standard** interpretation and play some role in the sequel. Non-standard interpretations form a generalization of standard interpretations, and as we will see below, this is a proper generalization. From this it trivially follows that every completeness result which holds for standard interpretations also holds for non-standard interpretations, but we have to show that soundness is preserved. This however is also straightforward, as there is only one rule involving $\top$ and it can be easily seen to be sound under non-standard interpretations.

This is how we interpret the types of our logic. What we want to interpret next is the *sequents* of the form $\Gamma \vdash \alpha$. We say that a sequent $R = \gamma_1, ..., \gamma_i \vdash \alpha$ is true in a model $\mathcal{M}$ under assignment σ, in symbols: $(\mathcal{M}, \sigma) \models \gamma_1, ..., \gamma_i \vdash \alpha$, if and only if $\overline{\sigma}(\gamma_1 \bullet ... \bullet \gamma_i) \leq \overline{\sigma}(\alpha)$ holds in $\mathcal{M}$. That is, we interpret the ‘,’, which denotes concatenation in sequents, as $\cdot$ in the model, and $\vdash$ as $\leq$. In the sequel, for Γ a sequence of types, we will often write $\overline{\sigma}(\Gamma)$ as an abbreviation, where we leave the former translation implicit. For the case of theorems, that is, derivable sequents with no antecedent, we have the following convention: $(\mathcal{M},\sigma) \models\ \vdash \alpha$, iff $1 \leq \overline{\sigma}(\alpha)$ in $\mathcal{M}$, where 1 is the unit element of $\mathcal{M}$ (note that this case does not arise in L).

More generally, for a given class of (bounded) residuated lattices (monoids, semigroups) $\mathfrak{C}$, we say that a sequent is *valid* in $\mathfrak{C}$, in symbols, $\mathfrak{C} \models \gamma_1, ..., \gamma_i \vdash \alpha$, if for all $\mathcal{M} \in \mathfrak{C}$ and all interpretations σ, $(\mathcal{M},\sigma) \models \gamma_1, ...\gamma_i \vdash \alpha$ (here we have to distinguish between standard and non-standard interpretations).

4 Completeness: Previous Results

There are a number of completeness results for the logics we have considered here. We will consider the most general ones, which will be important in the sequel.

Theorem 6 *For the class RM of residuated monoids, the class RL of residuated lattices, the class $RL_\bot$ of bounded residuated lattices,*

1. *$RM \models \Gamma \vdash \alpha$ if and only if $\Vdash_{L1} \Gamma \vdash \alpha$,*
2. *$RL \models \Gamma \vdash \alpha$ if and only if $\Vdash_{\boldsymbol{FL}} \Gamma \vdash \alpha$,*
3. *$RL_\bot \models \Gamma \vdash \alpha$ if and only if $\Vdash_{\boldsymbol{FL}_\bot} \Gamma \vdash \alpha$.*

For reference on theorem 6, see [1], [2], [9]. The proofs for the above completeness theorems usually proceed via the Lindenbaum-Tarski construction: we interpret primitive types as atomic terms modulo mutual derivability, and define $\sigma(\alpha) \leq \sigma(\beta)$ iff $\alpha \vdash \beta$. Then we can perform an induction over constructors to get the same for arbitrary formulas/terms. So there are quite simple completeness proofs for the general case. These completeness results can actually be strengthened to the *finite model property.* A logic, equipped with a class of models and interpretations, is said to have finite model property if it is complete in the finite; that is, theorem 6 remains valid if we restrict ourself to finite models. These results are highly non-trivial; for example, classical first-order logic fails to have finite model property.

Theorem 7 *1. $L1$ has finite model property;*
2. $\boldsymbol{FL}$ has finite model property;
3. $\boldsymbol{FL}_\bot$ has finite model property.

For the first claim, consider [8]; the second and third has been established by [16]. We want to establish soundness and completeness of the calculi with respect to the class of syntactic concept lattices and their reducts. The latter results are crucial to show that completeness holds if we restrict ourselves to languages over finite alphabets.

Soundness of interpretations into SCL follows from soundness direction of theorem 6, because SCL is just a particular class of bounded residuated lattices.

As $L, L1, \mathbf{FL}$ are fragments of $\mathbf{FL}_\bot$, we get the same result for $L, L1$ and $\mathbf{FL}$, considering the terms which contain only the operators which have a counterpart in the logic.

Let SCL_{L1} be the class of SCL reducts with $\{\circ, /, \backslash\}$, which specify a unit, and SCL_{FL} be the class of SCL reducts with operators $\{\circ, /, \backslash, \vee, \wedge\}$, that is, without the constants $\top$ and $\bot$.

Theorem 8 *(Completeness)*

1. *If $SCL_{L1} \models \Gamma \vdash \alpha$, then $\Vdash_{L1} \Gamma \vdash \alpha$;*
2. *if $SCL_{FL} \models \Gamma \vdash \alpha$, then $\Vdash_{\boldsymbol{FL}} \Gamma \vdash \alpha$;*
3. *if $SCL \models \Gamma \vdash \alpha$, then $\Vdash_{\boldsymbol{FL}_\bot} \Gamma \vdash \alpha$;*

The completeness proofs can be found in [19]. The proof shows that for any (bounded) residuated lattice (reduct) S, there is a language $L(S)$ such that S can be isomorphically embedded in $SCL(L(S))$. This embedding thus preserves validity in both directions, and thus completeness follows. The language $L(S)$ is constructed with the elements of S as underlying alphabet. By the finite model property, we can conclude that the result remains valid if we restrict ourselves to languages over finite alphabets: if S is finite, $L(S)$ is a language over a finite alphabet (though still infinite!). So theorem 8 also holds for languages over finite alphabets only.

5 SCL_n – Completeness via Embeddings

We now extend theorem 8 to the structures $SCL_n : n \in \mathbb{N} \cup \{\omega\}$ (henceforth: $\mathbb{N}_\omega$). Again, we proceed by showing that there is an isomorphic embedding from $SCL(L) \to SCL_n(L)$. To increase readability and avoid misunderstandings, in the following we let $\triangleright, \triangleleft, \circ$ denote operations in SCL, $\blacktriangleleft, \blacktriangleright, \odot$ denote the corresponding operations in SCL_n. This convention however only concerns this section! We will exemplify the embedding for SCL_2, but it is easy to see that this can be extended to any $n \in \mathbb{N}$.

Assume $M \in SCL(L)$. We take a map $\alpha : \wp(\Sigma^*) \to \wp(\Sigma^* \times \Sigma^*)$, which is defined as a lifting of $\alpha' : w \mapsto (w, \epsilon)$ to sets composed with closure, so we define $\alpha(M) = (\alpha'[M])^{\blacktriangleright\blacktriangleleft}$. The following are more or less immediate:

1. $\alpha(M) \in SCL_2(L)$;
2. if $w \in M$, then $(w, \epsilon) \in \alpha(M)$;
3. $\alpha'[M] \star \alpha'[N] = \alpha'[M \star N]$, for $\star \in \{\cdot, \cup, \cap\}$.

The third point ensures that α' is a homomorphism for sets and classical set-theoretic operations of languages. Moreover, it is easy to see that α' is a bijection. So α' is an isomorphic embedding from $(\wp(\Sigma^*), \cdot, \cup, \cap)$ to $(\wp(\Sigma^* \times \Sigma^*), \cdot, \cup, \cap)$ Note that all these points remain valid if we suitably extend α' to α'_n, with $\alpha'_n(w) = (w, \overbrace{\epsilon, ..., \epsilon}^{n\ times})$, with α_n defined accordingly.

This is quite obvious. It becomes much less obvious, if we switch our attention to α, that is, add the closure operation. The reason is as follows: $\alpha(M)$ might contain elements of the form (w, v) with $v \neq \epsilon$; and thus $\alpha(M) \cdot \alpha(N)$ might contain terms of the form $(w_1, w_2) \cdot (v_1, v_2) = (w_1 v_1, w_2 v_2)$. This is obviously problematic, as in $\alpha(M) \cdot \alpha(N)$ substrings occur in an order which differs from the one in $\mathsf{fo}(\alpha(M))\mathsf{fo}(\alpha(N))$. We show that the map α_n is nonetheless an isomorphic embedding $SCL(L) \to SCL_n(L)$. This requires some work, and we prove the claim step by step via the following lemmas (again, we exemplify this for $n = 2$, but results can be easily extended to the general case). We first make the following observation:

Lemma 9 $\alpha'[M]^{\blacktriangleright} = \{(x, y, z) : (x, yz) \in M^{\triangleright}\}$*;*

Both inclusions are obvious. This means that $\alpha(M)$ is the set of all (a, b) such that if $xMyz \subseteq L$, then $xaybz \in L$. This allows to show the following:

Lemma 10 *For $M = M^{\triangleright\triangleleft}$, $(w, \epsilon) \in \alpha(M)$ if and only if $w \in M$.*

Proof. *If*-direction is immediate. *Only if*: If $(w, \epsilon) \in \alpha(M)$, then whenever $(x, y, z) \in \alpha(M)^{\blacktriangleright}$, we have $xwyz \in L$. Now, $M^{\triangleright}$ is exactly the set of all (x, yz) such that $(x, y, z) \in \alpha(M)^{\blacktriangleright}$. Thus we have $w \in M^{\triangleright\triangleleft} = M$. ⊣

Put $M^a := \{wav : wv \in M\}$.

Lemma 11 $\alpha(M) \odot \alpha(N) \subseteq \alpha(M \circ N)$.

Proof. *Case 1*: Assume $(w, v) \in \alpha(M) \cdot \alpha(N)$. Then $(w, v) = (a_1 b_1, a_2 b_2)$, where $(a_1, a_2) \in \alpha(M)$, $(b_1, b_2) = \alpha(N)$. So for (a_1, a_2) it follows: if $xMyz \subseteq L$, then $xa_1ya_2z \in L$; the same holds for $(b_1, b_2) \in \alpha(N)$. So we have the following:

1. if $wMv \subseteq L$ $wa_1v^{a_2} \subseteq L$, and
2. if $wNv \subseteq L$, then $wb_1v^{b_2} \subseteq L$.

Case 1a: Assume there are x, y such that $xMNy \subseteq L$. Then it follows (by 2) that for every z_1, z_2 with $z_1 z_2 = y$, $xMb_1z_1b_2z_2 \subseteq L$, and consequently (by 1) that $xa_1b_1z_1a_2b_2z_2 \in L$, and so we have $(a_1b_1, a_2b_2) \in (\alpha'[M \circ N])^{\blacktriangleright\blacktriangleleft} = \alpha(M \circ N)$.

Case 1b: There are no x, y, z such that $xMNyz \subseteq L$. Then we have $M \circ N = \top$, and $MN^{\triangleright} = \emptyset$. By lemma 9, it follows that $\alpha'[MN]^{\blacktriangleright} = \emptyset$, and therefore, $\alpha'[MN]^{\blacktriangleright\blacktriangleleft} = \alpha(M \circ N) = \top$.

Case 2: $(w, v) \notin \alpha(M) \cdot \alpha(N)$. In this case, for all (x, y, z) such that for all $(a, b) \in \alpha(M) \cdot \alpha(N)$, $xaybz \in L$, we have $xwyvz \in L$. So it follows that if $(x, y, z) \in \alpha'[M \circ N]^{\blacktriangleright}$, then $xwyvz \in L$; thus $(w, v) \in \alpha'[M \circ N]^{\blacktriangleright\blacktriangleleft} = \alpha(M \circ N)$. ⊣

This is the first of a number of lemmas which establish the main theorem of this section. The second one establishes the inverse inclusion:

Lemma 12 $\alpha(M \circ N) \subseteq \alpha(M) \odot \alpha(N)$.

Proof. Assume $(w, v) \in \alpha(M \circ N)$.

Case 1: $(w, v) \in \alpha'[M \circ N]$. Then $v = \epsilon$, and $w \in M \circ N$. For all $w \in MN$, $(w, \epsilon) \in \alpha(M) \cdot \alpha(N)$. Furthermore, if $w \in MN^{\triangleright\triangleleft}$, then $(w, \epsilon) \in (\alpha(M) \cdot \alpha(N))^{\blacktriangleright\blacktriangleleft}$ by as simple argument using lemma 9. Consequently, $(w, v) = (w, \epsilon) \in \alpha(M) \odot \alpha(N)$.

Case 2: Assume $(w, v) \notin \alpha'[M \circ N]$. Consequently, it holds that if $(x, y, z) \in \alpha'[M \circ N]^{\blacktriangleright}$, then $xwyvz \in L$. As $\alpha'[M \circ N] \subseteq \alpha(M) \odot \alpha(N)$ (by case 1), we have $\alpha'[M \circ N]^{\blacktriangleright} \supseteq (\alpha(M) \odot \alpha(N))^{\blacktriangleright}$, and so if $(x, y, z) \in (\alpha(M) \odot \alpha(N))^{\blacktriangleright}$, then $xwyvz \in L$, and $(w, v) \in (\alpha(M) \odot \alpha(N))^{\blacktriangleright\blacktriangleleft} = \alpha(M) \odot \alpha(N)$. ⊣

So α is a homomorphism of $\circ$ (more generally, every α_n is a $\circ$-homomorphism). To show that it preserves meets and joins does not require a lot of work. In order to save one step, we directly prove the claim for infinite meets.

Lemma 13 $\bigwedge_{i \in I} \alpha(M_i) = \alpha(\bigwedge_{i \in I} M_i)$.

This proof is rather straightforward, as $\wedge$ equals $\cap$ in our case, a fact we will make use of.

Proof. $\subseteq$ Assume $(w, v) \in \alpha(M_i)$ for all $i \in I$. This means for all $i \in I$, if $x(M_i)yz \subseteq L$, then $xwyvz \in L$. We have $(x, yz) \in (\bigwedge_{i \in I} M_i)^{\triangleright}$ iff and only if $(x, y, z) \in \alpha'[\bigwedge_{i \in I} M_i]^{\blacktriangleright}$ (lemma 9). So if $(x, y, z) \in \alpha'[\bigwedge_{i \in I} M_i]^{\blacktriangleright}$, then $x(\bigcap_{i \in I} M_i)yz \subseteq L$, and so $xwyvz \in L$, and so $(w, v) \in \alpha'[\bigwedge_{i \in I} M_i]^{\blacktriangleright\blacktriangleleft} = \alpha(\bigwedge_{i \in I} M_i)$.

$\supseteq$ Assume $(w, v) \in \alpha(\bigwedge_{i \in I} M_i)$. Because $\alpha(X) = \alpha'[X]^{\blacktriangleright\blacktriangleleft}$, α' being a pointwise map on sets and $[-]^{\blacktriangleright\blacktriangleleft}$ being a closure operator, from $\bigwedge_{i \in I} M_i \subseteq M_j : j \in I$ it follows that $\alpha(\bigwedge_{i \in I} M_i) \subseteq \alpha(M_j)$; and so $\alpha(\bigwedge_{i \in I} M_i) \subseteq \bigcap_{i \in I} \alpha(M_i) = \bigwedge_{i \in I} \alpha(M_i)$. ⊣

Now we can use the fact that in a complete lattice, we can use meets to define joins (and vice versa). This allows us to derive the following:

Lemma 14 $\alpha(M) \vee \alpha(N) = \alpha(M \vee N)$.

Proof. We use the facts that 1. both $SCL(L), SCL_2(L)$ are complete, and 2. α preserves infinite meets. For these reasons, the following equality holds:

$$\alpha(M) \vee \alpha(N) = \bigwedge\{\alpha(X) : X \geq M, N\} = \alpha(\bigwedge\{X : X \geq M, N\}).$$

Moreover, we can easily extend this to the infinite case:

$$\bigvee_{i \in I} \alpha(M_i) = \bigwedge\{\alpha(X) : X \geq M_i : i \in I\} = \alpha(\bigwedge\{X : X \geq M_i : i \in I\}) = \alpha(\bigvee_{i \in I} M_i).$$

⊣

Again, this can be easily extended to any α_n, $n \in \mathbb{N}_\omega$.

Needless to say, every map $\alpha_n : SCL(L) \to SCL_n(L)$ is an injection. To see this, just assume we have $M, N \in SCL(L)$; and assume without loss of generality that $(w, v) \in M^{\triangleright}$, $(w, v) \notin N^{\triangleright}$. Then we have $(w, \epsilon, v) \in \alpha'[M]^{\blacktriangleright}$, but $(w, \epsilon, v) \notin \alpha'[N]^{\blacktriangleright}$, so $\alpha(M) = \alpha'[M]^{\blacktriangleright\blacktriangleleft} \neq \alpha'[N]^{\blacktriangleright\blacktriangleleft} = \alpha(N)$. This together with the with fact that we preserve joins and meets makes the following rather obvious:

Lemma 15 *$X \circ N \leq M$ iff $\alpha(X) \odot \alpha(N) \leq \alpha(M)$.*

Proof. *If*: For contraposition, assume $X \circ N \not\leq M$. Then there is $w \in X \circ N$, $w \notin M$. Consequently, there is $(w, \epsilon) \in \alpha(X) \odot \alpha(N)$, but $w \notin \alpha(M)$ (by lemma 10).

Only if: Assume $X \circ N \leq M$. Then obviously $\alpha(X \circ N) = \alpha(X) \odot \alpha(N) \leq \alpha(M)$, as α preserves $\subseteq$. $\dashv$

Lemma 16 $\alpha(M)/\alpha(N) = \alpha(M/N)$

Proof. We have $M/N = \bigvee\{X : X \circ N \leq M\}$; moreover $\alpha(\bigvee\{X : X \circ N \leq M\}) = \bigvee\{\alpha(X) : X \circ N \leq M\}$. Since $X \circ N \leq M$ iff $\alpha(X) \odot \alpha(N) \leq \alpha(M)$, we have $\bigvee\{\alpha(X) : X \circ N \leq M\} = \bigvee\{\alpha(X) : \alpha(X) \odot \alpha(N) \leq \alpha(M)\} = \alpha(M)/\alpha(N)$. $\dashv$

Again, this proof works perfectly fine for any α_n. To not get confused with $\bot, \top$ in SCL, SCL_2, we denote the latter elements with $\bot_2, \top_2$ etc.

Lemma 17 *$\alpha(\bot) = \bot_2$, but there are languages L such that $\alpha(\top) \neq \top_2$.*

Proof. 1. $\bot$ We have defined $\bot = \emptyset^{\triangleright\triangleleft}$. Assume $\bot = \emptyset$; in this case, the result is obvious. Assume there is $w \in \bot$. Then for every $(x, y) \in \Sigma^* \times \Sigma^*$, $xwy \in L$. Consequently, for all $(x, y, z) \in (\Sigma^*)^3$, $xwyz \in L$. So $\alpha'[\bot]^{\blacktriangleright} = \emptyset^{\blacktriangleright}$, and $\alpha(\bot) = \bot_2$.

2. Take the language $L = a(a+b)^*a$. Then $(a, a) \in ((a+b)^*)^{\triangleright}$, where $(a+b)^* = \top$. Consequently, $(a, a, \epsilon) \in \alpha'[\top]^{\blacktriangleright}$. As $abab \notin L$, we have $(b, b) \notin \alpha'[\top]^{\blacktriangleright\blacktriangleleft} = \alpha(\top)$, hence $\alpha(\top) \neq \top_2 = \Sigma^* \times \Sigma^*$. $\dashv$

Again, this is easily extended to arbitrary $n \in \mathbb{N}_\omega$. So we have a serious problem, because our embedding does not preserve $\top$. We can dodge this however by considering non-standard interpretations (see section 3.2 and proof of theorem 19 below).

So this proves the first main theorem:

Theorem 18 *For every $n \in \mathbb{N}_\omega$, there is an isomorphic embedding $\alpha_n : SCL(L) \to SCL_n(L)$, such that $\alpha_n(\bot) = \bot$, and which in addition preserves infinite meets and joins.*

From theorem 18 it is rather easy to extend the completeness result to SCL_n:

Theorem 19 *For arbitrary $n \in \mathbb{N}_\omega$, $\Vdash_{\boldsymbol{FL}_\bot} \Gamma \vdash \alpha$ iff $SCL_n \models \Gamma \vdash \alpha$.*

Proof. Soundness is clear and follows from more general results. Regarding completeness: Assume we have $\nVdash_{\mathbf{FL}_\bot} \Gamma \vdash \gamma$. Then there is an $L, \sigma : Pr \to SCL(L)$ such that $\overline{\sigma}(\Gamma) \not\subseteq \overline{\sigma}(\gamma)$. It follows that $\alpha_n(\overline{\sigma}(\Gamma) \not\subseteq \alpha_n(\overline{\sigma}(\gamma))$, and so $SCL_n(L), \alpha_n \circ \sigma \not\models \Gamma \vdash \gamma$, which proves the claim.

But keep in mind that $\alpha_n \circ \sigma$ is a non-standard interpretation, as $\alpha_n \circ \sigma(\top)$ need not be $\top$ in the lattice! $\dashv$

The results obviously also hold for the logics $\mathbf{FL}, L1$ and the corresponding syntactic concept lattice reducts (for these, the notions of standard and non-standard interpretation coincide). Note also that this shows that the notion of a non-standard interpretation properly generalizes standard interpretations.

6 A Characterization for Finite SCL_ω-structures

Obviously, if $L \notin Reg$, then $SCL_\omega(L)$ is infinite; but the converse is wrong (see lemma 2). A permutation π is a map on words which preserves all cardinalities of all letters in this word. For any language L, define $\Pi(L) = \{w : \pi(v) = w$ for some permutation π and some $v \in L\}$. Let $PermReg$ be the class of languages, which is defined as follows:

Definition 20 *$L \in PermReg$, iff 1. $L \in Reg$, and 2. $\Pi(L) = L$.*

This concerns, for example, languages like $\{w : |w|_a$ is even for $a \in \Sigma\}$. We will show that $SCL_\omega(L)$ is finite iff $L \in PermReg$. For the *if*-direction, we first show the following lemma, which at the same time gives some understanding of the combinatorics of permutations:

Lemma 21 *Assume $L \neq \Pi(L)$, so there is a permutation π, $w \in L$, such that $\pi(w) \notin L$. Then there are $\overline{w}, \overline{v} \in (\Sigma^*)^\omega$ such that $\mathsf{fo}(\overline{v} \cdot \overline{w}) = w$, $\mathsf{fo}(\overline{w})\mathsf{fo}(v) = \pi(w)$.*

Note firstly that assumptions assure that $w \neq \pi(w)$. From this follows that $\mathsf{fo}(\overline{w}) \neq \epsilon \neq \mathsf{fo}(\overline{v})$, as this would entail $w = \pi(w)$.

Proof. We choose some arbitrary w, π such that $w \in L, \pi(w) \notin L$ (which exist by assumption). We let a_i denote the ith letter of $\pi(w)$. We construct $\overline{w}$ in the following fashion:

Step 1 Take a_1, the first letter of $\pi(w)$, and put $\overline{w} = (a_1, \epsilon, \epsilon, ...)$. Of course, there is $\overline{v} \in (\Sigma^*)^\omega$ such that $\mathsf{fo}(\overline{v} \cdot (a_1, \epsilon, ...)) = w$, because a_1 occurs in some place in w. Now there are two possible cases:

Case 1: $\mathsf{fo}(\overline{w})\mathsf{fo}(\overline{v}) \notin L$; then we change the "target permutation" π from the lemma to ξ, where $\xi(w) = \mathsf{fo}(\overline{w})\mathsf{fo}(\overline{v})$ (this is clearly a permutation). Then we are done, as ξ, w satisfy the claim!

Case 2: $\mathsf{fo}(\overline{w})\mathsf{fo}(\overline{v}) \in L$. In this case, we discard w, π and consider w^1, π^1 instead, where $w^1 = \mathsf{fo}(\overline{w})\mathsf{fo}(\overline{v}) \in L$, and π^1 is defined by $\pi^1(w^1) = \pi(w)$ (this works because w, w^1 are permutations of each other). Then continue with step 2.

Step 2 Having chosen a_i before, we now take a_{i+1} (as $\pi^i(w^i) = \pi(w)$, it does not matter which of the two we consider). Put $\overline{w} = (a_1...a_i, a_{i+1}, \epsilon, ...)$; there is obviously a $\overline{v}$ such that $\mathsf{fo}((a_1...a_i, a_{i+1}, \epsilon, ...) \cdot \overline{v}) = w^i$, because w^i is constructed as $a_1...a_i v$, and v necessarily contains the letter a_{i+1}. Now we can go back to the case distinction and repeat the procedure.

In the end, there are two possibilities: as w is a finite word, either at some point we hit case 1, and the claim follows. Assume we do *not* hit case 1. Then at some point we have $i = |w| = |\pi(w)|$, so we construct $w^{|w|}$ as $a_1...a_{|w|}$. Then by definition and assumption, we have $a_1...a_{|w|} = \pi(w) \notin L$. But we also have, as we do not hit case 1 by assumption, $a_1...a_{|w|} = \mathsf{fo}(\overline{w})\mathsf{fo}(\overline{v}) = w^{|w|} \in L$ – contradiction. $\dashv$

As we can see, we can even make sure that $\overline{w}$ has the form $(w, a, \epsilon, \epsilon, ...)$, and $\overline{v} = (v_1, v_2, v_3, \epsilon, \epsilon, ...)$.

Lemma 22 *$L \in PermReg$ if and only if $SCL_\omega(L)$ is finite.*

Proof. *Only if*: There are only finitely many non-equivalent concepts of the form $(w, \epsilon, \epsilon, ...)$. Moreover, by permutation closure, we know that if $\mathsf{fo}(\overline{w}) = \mathsf{fo}(\overline{v})$, then $\{\overline{w}\}^\triangleright = \{\overline{v}\}^\triangleright$ and the claim follows easily.

If: For this we need the previous lemma. We prove the contraposition, so assume $L \notin PermReg$. Then either $L \notin Reg$, and the claim follows easily. Or $\Pi(L) \neq L$. In this case, we have w, π such that $w \in L$, $\pi(w) \notin L$, and there are $\overline{w}, \overline{v} \in (\Sigma^*)^\omega$ such that $\mathsf{fo}(\overline{v} \cdot \overline{w}) = w$, $\mathsf{fo}(\overline{w})\mathsf{fo}(v) = \pi(w)$. Moreover, $\overline{w} = (w, a, \epsilon, \epsilon, ...)$, and $\overline{v} = (v_1, v_2, v_3, \epsilon, \epsilon, ...)$.

Now for every $n \in \mathbb{N}$, we simply take a tuple $(\overbrace{\epsilon, ..., \epsilon}^{2n\ times}, w, \epsilon, \epsilon, ...)$. It is clear that for every n, we get non-equivalent tuples: We have

$$(\#)\ \mathsf{fo}((\epsilon_1, ..., \epsilon_{2n}, v_1, v_2, v_3, \epsilon, \epsilon, ...) \cdot (\epsilon_1, ..., \epsilon_{2(n-1)}, w, a, \epsilon, \epsilon, ...)) = \pi(w) \notin L,$$

whereas

$$\mathsf{fo}((\epsilon_1, ..., \epsilon_{2n}, v_1, v_2, v_3, \epsilon, \epsilon, ...) \cdot (\epsilon_1, ..., \epsilon_{2n}, w, a, \epsilon, \epsilon, ...)) = w \in L.$$

Moreover, (#) holds if in the term $2n$ is replaced by any number $m \geq 2n$. Put $\overline{w}_m = (\epsilon_1, ..., \epsilon_{2m}, w, a, \epsilon, \epsilon, ...)$. So for any $\overline{w}_m, \overline{w}_n$, if $m \neq n$, then $\{\overline{w}_m\}^\triangleright \neq \{\overline{w}_n\}^\triangleright$, and as these sets are closed and there are infinitely many of them, $SCL_\omega(L)$ is infinite. ⊣

7 Conclusion

We have shown completeness results for extensions of syntactic concepts to finite and infinite tuples; moreover, we have given a precise characterization of the class of languages which result in finite lattices in all cases. Interpreting substructural logics in sets of tuples rather than sets of strings is interesting for a number of reasons: from the perspective of categorial grammar and/or Lambek calculus as language-recognizing devices, the interpretation in tuples allows us to recognize languages which are not context-free (by letting grammars recognize tuples modulo fo). This relates more "classical" categorial approaches to new approaches such as the displacement calculus **D**, which also recognizes languages which are not context-free. In this context, infinite tuples are particularly interesting, as they allow to simulate both the "wrapping"-style extended concatenation in **D** and the "crossing"-style extended concatenation we have looked at in this paper. The usage of formal concept analysis is particularly interesting in connection with learning theory; so the results here might also be of some interest for learning beyond context-free languages.

References

1. Wojciech Buszkowski. Completeness results for Lambek syntactic calculus. *Mathematical Logic Quarterly*, 32(1-5):13–28, 1986.

2. Wojciech Buszkowski. Algebraic structures in categorial grammar. *Theor. Comput. Sci.*, 1998(1-2):5–24, 1998.
3. Alexander Clark. A learnable representation for syntax using residuated lattices. In Philippe de Groote, Markus Egg, and Laura Kallmeyer, editors, *Proceedings of the 14th Conference on Formal Grammar*, volume 5591 of *Lecture Notes in Computer Science*, pages 183–198. Springer, 2009.
4. Alexander Clark. Learning context free grammars with the syntactic concept lattice. In José M. Sempere and Pedro García, editors, *10th International Colloquium on Grammatical Inference*, volume 6339 of *Lecture Notes in Computer Science*, pages 38–51. Springer, 2010.
5. Alexander Clark. Logical grammars, logical theories. In Denis Béchet and Alexander Ja. Dikovsky, editors, *LACL*, volume 7351 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2012.
6. Alexander Clark. The syntactic concept lattice: Another algebraic theory of the context-free languages? *Journal of Logic and Computation*, 2013.
7. B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order.* Cambridge University Press, Cambridge, 2 edition, 1991.
8. Maciej Farulewski. On finite models of the lambek calculus. *Studia Logica*, 80(1):63–74, 2005.
9. Nikolaos Galatos, Peter Jipsen, Tomasz Kowalski, and Hiroakira Ono. *Residuated Lattices: An Algebraic Glimpse at Substructural Logics.* Elsevier, 2007.
10. Zellig S. Harris. *Structural Linguistics.* The University of Chicago Press, 1963.
11. Makoto Kanazawa. The Lambek calculus enriched with additional connectives. *Journal of Logic, Language, and Information*, 1:141–171, 1992.
12. Makoto Kanazawa, Jens Michaelis, Sylvain Salvati, and Ryo Yoshinaka. Well-nestedness properly subsumes strict derivational minimalism. In *Logical Aspects of Computational Linguistics - 6th International Conference, LACL 2011, Montpellier, France, June 29 - July 1, 2011. Proceedings*, pages 112–128, 2011.
13. Joachim Lambek. The Mathematics of Sentence Structure. *The American Mathematical Monthly*, 65:154–169, 1958.
14. Joachim Lambek. On the calculus of syntactic types. In Roman Jakobson, editor, *Structure of Language and its Mathematical Aspects*, pages 166–178. Providence, 1961.
15. Glyn Morrill, Oriol Valentín, and Mario Fadda. The displacement calculus. *Journal of Logic, Language and Information*, 20(1):1–48, 2011.
16. Mitsuhiro Okada and Kazushige Terui. The finite model property for various fragments of intuitionistic linear logic. *J. Symb. Log.*, 64(2):790–802, 1999.
17. M. Pentus. Lambek grammars are context free. In *Proceedings of the 8th Annual IEEE Symposium on Logic in Computer Science*, pages 429–433, Los Alamitos, California, 1993. IEEE Computer Society Press.
18. A. Sestier. Contributions à une théorie ensembliste des classifications linguistiques. (Contributions to a set–theoretical theory of classifications). In *Actes du Ier Congrès de l'AFCAL*, pages 293–305, Grenoble, 1960.
19. Christian Wurm. Completeness of full lambek calculus for syntactic concept lattices. In *Formal Grammar - 17th and 18th International Conferences, FG 2012, Opole, Poland, August 2012, Revised Selected Papers, FG 2013, Düsseldorf, Germany, August 2013. Proceedings*, pages 126–141, 2012.