Language-theoretic and Finite Relation Models for the (Full) Lambek Calculus

Christian Wurm
cwurm@phil.uni-duesseldorf.de
Heinrich Heine Universität Düsseldorf

March 22, 2017

Abstract

We prove completeness for some language-theoretic models of the Full Lambek calculus and its various fragments. First we consider syntactic concepts and syntactic concepts over regular languages, which provide a complete semantics for the full Lambek calculus $\mathbf{FL}_\bot$. We present a new semantics we call automata-theoretic, which combines languages and relations via closure operators which are based on automaton transitions. We establish the completeness of this semantics for the full Lambek calculus via an isomorphism theorem for the syntactic concepts lattice of a language and a construction for the universal automaton recognizing the same language. Finally, we use automata-theoretic semantics to prove completeness of relation models of binary relations and finite relation models for the Lambek calculus without and with empty antecedents (henceforth: **L** and **L1**), thus solving a problem left open by **(author?)** [24].

1 Introduction

The main contributions of this article are the following: we extend some established completeness results for the Full Lambek calculus $\mathbf{FL}_\bot$ and its fragments for syntactic concept lattices (SCL), the main point being the restriction to regular languages. (Throughout this article, we use completeness in the sense of completeness with respect to the internal consequence relation, that is, the semantics models the set of sequents which are derivable via the logical calculus – this distinguishes it from the approach in [1], who consider the external consequence relation; see [12] for the relation of the two). We provide a new kind of semantics we call "automata-theoretic", where closure operators relate strings with transitions they induce in an automaton (we call the resulting structure *automatic concept lattice*). We prove completeness for $\mathbf{FL}_\bot$ by showing that the syntactic concept lattice of a language is isomorphic to the automatic concept lattice of the universal automaton recognizing the same language (for the

universal automaton, consider [19], also [9]). As automata relate languages to relations, we can use this as point of departure to prove completeness of canonical relational semantics (by which we mean that formulas are interpreted as binary relations and $\bullet$ as relation composition) for $\mathbf{L}, \mathbf{L1}$ (Lambek calculus without and with empty antecedents). Since regularity of languages corresponds to finiteness of automata, completeness for regular languages allows us to extend this to completeness of finite relation models for $\mathbf{L}, \mathbf{L1}$, thus solving a problem left open by **(author?)** [24].

A note on the history of the problems we address: whereas the semantics of $\mathbf{FL}_\bot$ with languages and associated closure operators is rather recent [see 28], canonical relational semantics (as canonical L-models with $\bullet$ being interpreted as concatenation) has a long history: it is discussed in detail by **(author?)** [27] (but goes back much further), at which point completeness for both was still an open question. Completeness of L-models for the $\bullet$-free fragment of $\mathbf{L}$ was proved by **(author?)** [3], for $\mathbf{L}$ (and ϵ-free languages) by **(author?)** [24]. The "canonical" relational semantics of binary relations and composition is usually associated with a "dynamic" interpretation of formulas as computations in programs [see 27]. Completeness of this relational semantics has been shown by **(author?)** [2], who use relational quantales and prove results for a wide variety of substructural logics (or put differently, non-commutative linear logics). This approach differs from [1] in that it models internal consequences of the logic, as we do. An even stronger result has been obtained by **(author?)** [24]. Importantly, the "canonical" relational semantics of binary relations and composition has to be distinguished from semantics based on ternary relations, which is more general, see [4] for background and [25] for a general introduction. We have two interesting results regarding relational semantics: firstly, our automata-theoretic semantics shows how we can directly link language and relation models via a Galois connection, and secondly, we prove completeness of $\mathbf{L}, \mathbf{L1}$ for finite relation models, that is, finite algebras of finite relations.

And a short note on terminology: as in this article, the main focus will be on relational and language-theoretic models with and without closure operators, where the semantics without closure operators is rather well-established and well-known, the semantics with closure operators is rather recent. We therefore use the term "canonical" to refer to models not involving closure operators (we will explain this in more detail below. We are aware that this notion is used in different senses in the literature, still we do not have a better one.

The interesting thing about our approach is that most of the proofs follow a common pattern: we only provide one basic completeness proof, the one for syntactic concept lattices. The proof for concept lattices over regular languages and automata-theoretic semantics can be obtained via suitable embeddings/morphisms. To obtain the completeness results for canonical relation models, we use the following idea: we show that we can get rid of closure operators from automatic concepts while preserving truth and falsity of inequations (only for models of $\mathbf{L}, \mathbf{L1}$), while this construction preserves the property of finiteness of relations. So we have a variety of results with a rather small set of "serious" proofs.

The completeness proofs for $\mathbf{FL}_\bot$ etc. and syntactic concept lattices are already published [see 28], and so are the ones for automatic concept lattices [see 29]; we repeat the proofs because firstly there are some minor modifications, secondly they have some important properties to which we refer later on, and thirdly this will make the article self-contained. The central notion of this paper is the one of a **nucleus**, that is a closure operator γ on an ordered monoid/semigroup such that $\gamma(a) \cdot \gamma(b) \leq \gamma(a \cdot b)$. Whereas nuclei are already a central notion in most work on residuated lattices [see 12], for us there is a new focus: namely whether and under which circumstances we can get rid of them while preserving truth and falsity of inequations. In a sense, these are the main results of the paper and the key to "canonical" completeness result, which is in general the hardest to obtain.

The article is structured as follows: sections 2 to 5 will cover semantics for $\mathbf{FL}_\bot$ with models based on closure operators and Galois connections, with the results mostly being established (with some important exceptions). Section 6 provides an overview by establishing some general facts about closure operators on residuated lattices, and under which circumstances we can (not) get rid of them (in general). Section 7 provides the proof of completeness for finite relation models, which is based on automatic concepts.

2 The Logics L, L1, FL, $\mathbf{FL}_\bot$ and their Models

2.1 The Logics L, L1, FL and $\mathbf{FL}_\bot$

The Lambek calculus **L** was introduced by **(author?)** [16]. **L1** is a proper extension of **L**, and $\mathbf{FL}, \mathbf{FL}_\bot$ 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, $C_\mathbf{L} := \{/, \backslash, \bullet\}$, 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:

1. $Pr \subseteq Tp_C(Pr)$.
2. 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 also sometimes say "**L1** formula" for $Tp_{C_\mathbf{L}}(Pr)$, **FL**-formula for $Tp_{C_\mathbf{FL}}(Pr)$ etc., as it is more intuitive. 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; besides this, the calculi are identical. In **FL** and $\mathbf{FL}_\bot$ we also allow for empty sequents. Lowercase Greek letters range over single types. Below, we present the standard rules of the Lambek calculus **L** (and **L1**).

$$(ax) \qquad \alpha \vdash \alpha$$

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

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

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

These are the standard rules of **L** and **L1** (roughly as in [16]). 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 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 [17], and have been subsequently studied by **(author?)** [14] and **(author?)** [15].

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

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

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

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

This gives us the logic **FL**; if we discard $\wedge$ and the corresponding rules, we have the logic **L1**($\vee$), which is **L1** with additional connective $\vee$ (and the corresponding rules). Note that this slightly deviates from standard terminology, because usually, **FL** has an additional constant 0. 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.[1]

[1] Whereas **L** and **L1** are equally powerful in the sense of languages which are recognizable – except for languages containing the empty word, which cannot be recognized by **L** under standard assumptions – , **(author?)** [14] shows that **FL** is considerably more powerful than **L**: whereas **L** only recognizes context-free languages by the classical result of **(author?)** [23], **FL** can recognize any finite intersection of context-free languages.

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

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 **(author?)** [16], namely admissibility of the cut-rule:

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

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

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

For a general introduction and background to the structures we introduce here, see [12]. The standard model for **L** is the class of residuated semigroups. These are structures $(M, \cdot, \backslash, /, \leq)$, where $(M, \cdot)$ is a semigroup, $(M, \leq)$ is a partial order, and $\cdot, /, \backslash$ satisfy the law of residuation: for $m, n, o \in M$,

$$\text{(Res)}\ m \leq o/n \Leftrightarrow m \cdot n \leq o \Leftrightarrow n \leq m\backslash o.$$

Note that this implies that $\cdot$ respects the order $\leq$. The standard model for **L1** is the class of residuated monoids, which are structures $(M, \cdot, \backslash, /, 1, \leq)$ such that $(M, \cdot, \backslash, /, \leq)$ is a residuated semigroup and 1 is the neutral element for $\cdot$. The standard model for **FL** is the class of residuated lattices, and for $\mathbf{FL}_\bot$, the class of bounded residuated lattices A residuated lattice is a structure $(M, \cdot, \vee, \wedge, \backslash, /, 1)$, 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 $a \vee b = b$. At some points, we will also consider structures $(M, \cdot, \vee, \backslash, /, 1)$, that is, residuated monoids with least upper bound, but without greatest lower bound. A bounded residuated lattice is a structure $(M, \cdot, \vee, \wedge, \backslash, /, 1, \top, \bot)$, where $(M, \cdot, \vee, \wedge, \backslash, /, 1)$ is a residuated lattice, $\top$ is the maximal element of the lattice order $\leq$ and $\bot$ is its minimal element.

We call the class of residuated semigroups RS, the residuated monoids RM, the class of residuated monoids with least upper bound '$\vee$' $RM(\vee)$, 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 as elements of the algebra, and extend σ to $\overline{\sigma}$ by defining it appropriately for $1, \top, \bot$, and extending it inductively over our type constructors $C := \{/, \backslash, \bullet, \vee, \wedge\}$. 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. 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 that for all $\alpha \in Tp_C(Pr)$, $\overline{\sigma}(\alpha) \leq m$.

3. $\overline{\sigma}(\bot) = \bot$

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

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 $\top, \bot$: one which interprets them as the upper/lower bound of the lattice, which is the standard interpretation, and one which just interprets them as arbitrary elements which only have to be larger/smaller than the interpretation of any other formula. The latter will be called the **non-standard** interpretation and play some role in the sequel (but only for technical reasons). Every standard interpretation is a non-standard interpretation, but not conversely, as we will see below. From this it trivially follows that every completeness result which holds for standard interpretations also holds for non-standard interpretations (but not conversely); soundness of nonstandard semantics can be easily shown by the usual argument (induction over proof rules).

This is how we interpret the formulas of our logic. What we interpret next is the *sequents* of the form $\Gamma \vdash \alpha$. We say that a sequent $\gamma_1, ..., \gamma_i \vdash \alpha$ 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 $\mathcal{M}$. So 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 $1 \leq \overline{\sigma}(\alpha)$, 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).

2.3 Canonical Relation Models

We now define the basic operations for relations; assume we have a carrier set Q such that $R, R' \subseteq Q \times Q$. We have

$$R;R' = \{(x,y) : \exists z : (x,z) \in R, (z,y) \in R'\}$$

We refer to this as **relation composition**; residuals are defined by

$$R/R' = \{(x,y) \in Q \times Q : \{(x,y)\};R' \subseteq R\}$$
$$R'\backslash R = \{(x,y) \in Q \times Q : R';\{(x,y)\} \subseteq R\}$$

This definition is the unique one ensuring $;, /, \backslash$ satisfy (Res), given $Q \times Q$. Note however that the specification of the carrier set is necessary to make residuation uniquely defined!

Definition 1 *Let $\mathbf{R} \subseteq \wp(Q \times Q)$ be a set of relations such that if $R, R' \in \boldsymbol{R}$, then so are $R \cap R', R \cup R', R;R', R/R', R\backslash R'$, and let 1_R be such that for all $R \in \boldsymbol{R}$, $1_R;R = R;1_{\boldsymbol{R}} = R$. Then the structure $(\mathbf{R}, ;, \cup, \cap, /, \backslash, 1_{\mathbf{R}})$ is a residuated lattice which we call a (canonical) **algebra of relations**.*

The reducts of this algebra are defined accordingly. In general, we refer to Q as the **carrier set** of the algebra of relations. Given a class of algebras $\mathfrak{C} \in \{RL, RM(\vee), RM\}$, we will denote the corresponding class of canonical relation models by $\mathsf{Rel}(\mathfrak{C})$, and given **FL** or a fragment thereof, we call its (canonical) **relation models** the class of (canonical) algebras of relations of the appropriate signature. There is a "universal" unit relation (corresponding to $\{\epsilon\}$ in L-models), namely the identity relation on the carrier set $\mathsf{id}_Q = \{(x,x) : x \in Q\}$. Note that we do *not* require that $1_{\mathbf{R}} = \mathsf{id}_Q$, as depending on $\mathbf{R}$, there might other relations satisfying the criterion. The definition of $\mathsf{Rel}(RS)$, the relational residuated semigroups, is somewhat more involved, and simply skipping the unit relation is not sufficient. The reason is the following: for all relation models so far, $R \in \mathbf{R}$, we have $\mathsf{id}_Q \subseteq R/R, R\backslash R$, and hence for all $R, R' \in \mathbf{R}$, $R' \subseteq R';(R/R)$ etc. This does not hold for residuated semigroups in general and is not derivable in **L**. The problem is that **L** does not have theorems, so there is no logical counterpart for $\mathsf{id}_Q \subseteq R$; hence the algebra of relations would be more "expressive" than the logic, and completeness would fail. Note that in **L1** and its extensions, we do have theorems, so for these logics this does not pose any problems. For **L** and $\mathsf{Rel}(RS)$, we need to restrict (or, depending on perspective, generalize) the residuals in an appropriate fashion. We do this by fixing a **carrier relation** R_{max}, where we require that for all $R \in \mathbf{R}$, $R \subseteq R_{max}$, and we define operation $/_{max}, \backslash_{max}$ by

$$R/_{max}R' = \{(x,y) \in R_{max} : \{(x,y)\};R' \subseteq R\}$$
$$R'\backslash_{max}R = \{(x,y) \in R_{max} : R';\{(x,y)\} \subseteq R\}$$

Note that this is a generalization of the residuation as defined above, as in the above case, we simply have $R_{max} = Q \times Q$. In the general case, $R' \subseteq R';R/_{max}R$ etc. does not necessarily hold. Besides, we obviously require that $\mathbf{R}$ is closed

under these operations, and so a relational residuated semigroup is a structure $(\mathbf{R}, ;, /_{max}, \backslash_{max}, \subseteq)$ satisfying the usual laws. We denote the class of these structures by $\mathsf{Rel}(RS)$.

By $\mathsf{FinRel}(\mathfrak{C})$, where $\mathfrak{C} \in \{RL, RM(\vee), RM, RS\}$, we denote the class of *strictly finite* relation models, that is, $\mathsf{FinRel}(\mathfrak{C})$ is the subclass of $REL(\mathfrak{C})$ of algebras based on a set of relations $\mathbf{R}$ such that 1. for all $R \in \mathbf{R}$, R is finite, and 2. $\mathbf{R}$ is finite. So $\mathsf{FinRel}(\mathfrak{C})$ is the class of finite algebras (of the appropriate type) of finite relations. Note that there can be infinite algebras of finite relations (and vice versa), but given a finite set $\mathbf{R}$ of finite relations, we can always fix a finite carrier set such that the algebra of relations generated by $\mathbf{R}$ will be finite.

For $\cap, \cup$ we have the obvious set-theoretic equalities, in particular the following:

$$\text{(distr) } R_1 \cap (R_2 \cup R_3) = (R_1 \cap R_2) \cup (R_1 \cap R_3)$$

We have $\Vdash_{\mathbf{FL}} (\alpha \wedge \beta) \vee (\alpha \wedge \gamma) \vdash \alpha \wedge (\beta \vee \gamma)$, but $\nVdash_{\mathbf{FL}} \alpha \wedge (\beta \vee \gamma) \vdash (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$. So as one inequality is not derivable in **FL**, there is no hope for completeness of $\mathsf{Rel}(RL)$ for **FL**. In principle, we might have completeness for $\mathbf{L1}(\vee)$; but we leave this as an open question (this problem is addressed in [2], but the authors do not interpret the logical $\vee$ as $\cup$, but rather as involving a closure operator). Completeness of $\mathsf{Rel}(RM), \mathsf{Rel}(RS)$ for $\mathbf{L1}, \mathbf{L}$ has been shown by **(author?)** [2] and **(author?)** [24]. We will strengthen this to $\mathsf{FinRel}(RM), \mathsf{FinRel}(RS)$.

"Canonical" models for the Lambek calculus, such as language (or L-) models and canonical relation models have a peculiar property: to know the result of an operation, we only need to know the operands, not the underlying algebra. It is in this sense that these models are *canonical* (note that this holds only in a restricted sense for $\mathsf{Rel}(RS)$, as we have to additionally specify R_{max}). This also allows to simply skip the underlying model and just specify the atomic interpretation and the fact that it is extended to a certain canonical model. There is another important property of canonical semantics (consequence of the one mentioned here), namely that they can be conceived of as *frame semantics* rather than algebraic semantics, as we do here. For reasons of space, we just mention this possibility, and refer the reader to the literature for more background [see 25, 1].

2.4 Syntactic Concepts and Galois Connections

We now present a less well-known semantics for $\mathbf{FL}_{\bot}$, which is language-theoretic and based on closure operators. Syntactic concept lattices form a particular case of what is well-known as formal concept lattice (or formal concept analysis) in computer science [see 10]. In linguistics, they have been introduced by **(author?)** [26]. They were brought back to attention and enriched with a monoid operation and residuation by **(author?)** [6] [see also 9], as they turn out to be useful representations for language learning [see 7, 18].

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 (see, e.g. [13]). 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 $w \sim_L v$, 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, with $(w_1, v_1) \sim_L^2 (w_2, v_2)$, iff for all $x, y, z \in \Sigma^*$, $xw_1yv_1z \in L \Leftrightarrow xw_2yv_2z \in L$, etc., but we will not consider this more general case here [see 30]. A 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 (finite) dataset; then it is quite improbable that we get the equivalence classes we would usually desire. Syntactic concepts provide a somewhat less rigid notion of equivalence, which can be conceived of as equivalence restricted to a given set of relevant contexts (we will specify this below). This at least partly overcomes the difficulties we have mentioned here.

Given a language $L \subseteq \Sigma^*$, we define two maps: a map $[-]^{\triangleright} : \wp(\Sigma^*) \to \wp((\Sigma^*)^2)$, and $[-]^{\triangleleft} : \wp((\Sigma^*)^2) \to \wp(\Sigma^*)$, which are defined as follows:

$$\text{for } M \subseteq \Sigma^*,\ M^{\triangleright} := \{(x, y) : \forall w \in M, xwy \in L\} \tag{1}$$

$$\text{for } C \subseteq (\Sigma^*)^2,\ C^{\triangleleft} := \{w : \forall (x, y) \in C, xwy \in L\} \tag{2}$$

We refer to the domain of $[-]^{\triangleright}$ as the **extent**, its co-domain as the **intent**. In our case, the intent is a set of strings which is mapped to the extent, which is the set of context-tuples 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. The maps $[-]^{\triangleright}, [-]^{\triangleleft}$ form a Galois connection, see [see 6, 28]. For extension of these maps to larger tuples, consider [8, 21, 30]. Importantly, all these are special applications of the general theory of Galois connections; for background, see [12]. Obviously, $[-]^{\triangleleft}$ and $[-]^{\triangleright}$ are only defined with respect to a given language L, otherwise they are meaningless. As long as it is clear about which language (if any particular language) 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). Hence syntactic concepts allow us to restrict distributional equivalence to certain context sets, namely exactly those which are closed under $[-]^{\triangleleft\triangleright}$.

Regardless of the underlying objects, the two compositions of the maps, $[-]^{\triangleleft\triangleright}$ and $[-]^{\triangleright\triangleleft}$, form a **closure operator**, 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 $\triangleleft\triangleright$. Note also that for any set of strings M and contexts C, $M^{\triangleright} = M^{\triangleright\triangleleft\triangleright}$ and $C^{\triangleleft} = C^{\triangleleft\triangleright\triangleleft}$. A set M is **closed**, if $M^{\triangleright\triangleleft} = M$ etc. The closure operator $[-]^{\triangleright\triangleleft}$ gives rise to a lattice $(\mathcal{B}_L, \leq)$, where the elements of $\mathcal{B}_L$ are the sets $M \subseteq \Sigma^*$ 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 an antitone **Galois connection** between the two (see [10] for more background), that is:

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

The concept of a Galois connection is in fact one of the central notions of this article. 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 (S, C), where $S^{\triangleright} = C$ and $C^{\triangleleft} = S$. For our purposes, we mostly need to consider only the left component, so we suppress the contexts and only consider the stringsets of the form $M^{\triangleright\triangleleft}$. An exception to this convention is section 5, where we will make use of concepts as pairs (M, C) with $M = C^{\triangleleft}$, $C = M^{\triangleright}$, as it will increase readability in this case. 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 (with the exception of section 5), the concept in the classical sense being easily reconstructible.

Definition 2 *For $[-]^{\triangleright\triangleleft}$ defined with respect to $L \subseteq \Sigma^*$, let $\mathcal{B}_L$ denote the set of $[-]^{\triangleright\triangleleft}$-closed subsets of Σ^*. This set forms a bounded lattice $(\mathcal{B}_L, \wedge, \vee, \top, \bot)$, where $\top = \Sigma^*$, $\bot = \emptyset^{\triangleright\triangleleft}$, and for $M, N \in \mathcal{B}_L$, $M \wedge N = M \cap N$, $M \vee N = (M \cup N)^{\triangleright\triangleleft}$.*

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 also easy to verify that this forms a complete lattice, as infinite joins are defined by (closure of) infinite unions, infinite meets by infinite intersections.

We will also consider another set of polar maps, where everything remains the same, except for the fact that the extents are subsets of Σ^+ rather than Σ^*; we thus ban ϵ from closed stringsets. We denote the maps by $[-]^{+\triangleright}$, $[-]^{+\triangleleft}$, and closed sets are sets $M \subseteq \Sigma^+$ such that $M = M^{+\triangleright+\triangleleft}$. We refer to these as **positive concepts** and we denote the set of positive concepts of a language with $\mathcal{B}_L^+$. It is easy to see that this set forms a complete bounded lattice as well, so we can use the same operators as for $\mathcal{B}_L$.

2.5 Monoid Structure and Residuation for Syntactic Concepts

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 $M \circ N := (M \cdot N)^{\triangleright\triangleleft}$, where $M \cdot N = \{wv : w \in M, v \in N\}$. We usually write MN for $M \cdot N$, if M, N are sets of strings. '$\circ$' is associative on concepts: For $M, N, O \in \mathcal{B}_L$, $M \circ (N \circ O) = (M \circ N) \circ O$. This follows from the fact that $[-]^{\triangleright\triangleleft}$ is a **nucleus**, that is, it is a closure operator and in addition it satisfies $M^{\triangleright\triangleleft}N^{\triangleright\triangleleft} \subseteq (MN)^{\triangleright\triangleleft}$, and the associativity of $\cdot$-concatenation.

Furthermore, it is easy to see that the neutral element of '$\circ$' is $\{\epsilon\}^{\triangleright\triangleleft}$ (which need not be $\{\epsilon\}$). The monoid operation respects the partial order of the lattice, that is, for $X, Y, Z, W \in \mathcal{B}_L$, 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

$$\bigvee_{Z \in \mathbf{Z}} (X \circ Z \circ Y) = X \circ \left(\bigvee \mathbf{Z}\right) \circ Y$$

Here $\leq$ follows algebraically ($\circ$ respects the order), and $\geq$ follows from the fact that 1. $\bigcup$ distributes over $\cdot$ (infinite unions distribute over concatenation), and 2. $[-]^{\triangleright\triangleleft}$ is a nucleus. We can thus also conceive of syntactic concepts with $\bigvee, \circ, 1$ as **quantales** (see for example [31], by now a huge literature on quantales, particularly interesting in connection with this article is [2]), but we will not consider quantales in this article. However, we use these facts for 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. It is a well-known fact (and easy to prove) that for any complete lattice with a monoid operation distributing over infinite joins, residuals defined as above, we have $Y \leq X\backslash Z$ iff $X \circ Y \leq Z$ iff $X \leq Z/Y$, hence residuals satisfy (Res). So every quantale and every complete lattice with a monoid operation based on a nucleus can be extended to a residuated lattice. We can also explicitly define the residuals as **(author?)** [6] by putting $X/Y = \{(x, Yy) : (x, y) \in X^{\triangleright}\}^{\triangleleft}$, $Y\backslash X = \{(xY, y) : (x, y) \in X^{\triangleright}\}^{\triangleleft}$. It is easy to check that the definitions coincide.

Definition 4 *The **syntactic concept lattice** of a language L is defined as $SCL(L) := (\mathcal{B}_L, \circ, \wedge, \vee, /, \backslash, 1, \top, \bot)$, where $\mathcal{B}_L, \wedge, \vee, \top, \bot$ are defined as in Definition 2, $1 = \{\epsilon\}^{\triangleright\triangleleft}$, and $\circ, /, \backslash$ are as defined above.*

We will denote by SCL the class of all lattices of the form $SCL(L)$ for some language L, without any further requirement regarding L. We can apply the definition of interpretations to the case of SCL, so it is clear how $\mathbf{FL}_{\bot}$ is interpreted in SCL.

Recall that a reduct of an algebra is the same algebra with only a proper subset of connectives; the notion easily extends to classes. We let $SCL_{\mathbf{FL}}$ be the class of SCL reducts with operators $\{\circ, /, \backslash, \vee, \wedge\}$ without the constants $\top$ and $\bot$, $SCL_{\mathbf{L1}(\vee)}$ the reduct to $\{\vee, \circ, /, \backslash\}$ and $SCL_{\mathbf{L1}}$ be the class of SCL reducts with $\{\circ, /, \backslash\}$, which all specify a unit. We now apply the same notions to positive concepts:

Definition 5 *The **positive syntactic concept lattice** of a language L is defined as $SCL^{+}(L) := (\mathcal{B}_L^{+}, \circ, \wedge, \vee, /, \backslash, \top, \bot)$, where $\mathcal{B}_L^{+}, \wedge, \vee, \top, \bot$ $\circ, /, \backslash$ are all as defined above.*

Note that the structure is very similar, yet it does not have a 1, that is, a neutral element for $\circ$. Definition 3 works equally well in this case, making sure that there is no ϵ in closed sets. By SCL^+ we denote the class of all structures $SCL^+(L)$ for an arbitrary language L; by $SCL^+_{\mathbf{L}}$ the class of all $\{\circ, /, \backslash\}$ reducts of SCL^+ without any constants. So it is clear how the logical fragments $\mathbf{FL}, \mathbf{L1}(\vee), \mathbf{L1}$ are interpreted in the reducts of SCL to $\{\vee, \wedge, 1, /, \backslash, \circ\}$, $\{\vee, 1, \backslash, /, \circ\}$ etc., because these reducts are residuated lattices, monoids etc. As regards $\mathbf{L}$, it can also be interpreted in an SCL reduced to a semigroup; that will however be an incomplete semantics, because sequents such as $p \vdash p \bullet (q/q)$ are not derivable in $\mathbf{L}$, but valid in the semigroup-reduct of SCL. Instead, we interpret $\mathbf{L}$ in the residuated semigroup-reduct $SCL^+_{\mathbf{L}}$ of positive syntactic concept lattices.

Syntactic concepts are related to an order which will have some importance in the sequel. Given a language $L \subseteq \Sigma^*$, we write $w \leq_L v$ iff $xvy \in L \Rightarrow xwy \in L$. Note that $\leq_L$ is a pre-order, as from $w \leq_L v$ and $v \leq_L w$ follows $w \sim_L v$, where $\sim_L$ is substitutional equivalence, not equality. As is easy to see, $\leq_L$ respects concatenation of strings. We say a set (of strings) W is **downward closed** (with respect to $\leq_L$), if from $w \in W$ and $v \leq_L w$ it follows that $v \in W$. Obviously, given a language $L \subseteq \Sigma^*$ and a set of strings $W \subseteq \Sigma^*$, if $W = W^{\triangleright\triangleleft}$, then W is downward closed w.r.t. $\leq_L$. We can also extend $\leq_L$ to sets: we write $N \leq_L M$, if $xMy \subseteq L \Rightarrow xNy \in L$. Note that $N \leq_L M$ iff for all $w \in N$, $w \leq_L M$, but this resolution to strings does not work for M (the right-hand side). Moreover, it is easy to see that $N \leq_L M$ iff $N^{\triangleright\triangleleft} \subseteq M^{\triangleright\triangleleft}$.

2.6 Completeness: Previous Results

There are a number of well-established completeness results for the logics we have considered here. We quickly present the ones which will be important in the sequel.

Theorem 1 *1.* $RS \models \Gamma \vdash \alpha$ *if and only if* $\Vdash_{\boldsymbol{L}} \Gamma \vdash \alpha$

2. $RM \models \Gamma \vdash \alpha$ *if and only if* $\Vdash_{\boldsymbol{L1}} \Gamma \vdash \alpha$

3. $RL \models \Gamma \vdash \alpha$ *if and only if* $\Vdash_{\boldsymbol{FL}} \Gamma \vdash \alpha$

4. $RL_\bot \models \Gamma \vdash \alpha$ *if and only if* $\Vdash_{\boldsymbol{FL}_\bot} \Gamma \vdash \alpha$

For reference on Theorem 1, see [4, 5, 12]. 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 $\alpha \vdash \beta$. One 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 the finite model property if it is complete in the finite; that is, Theorem 1 remains valid if we restrict ourselves to finite models.

These results are highly non-trivial; for example, classical first-order logic fails to have finite model property.

Theorem 2 *1.* $\boldsymbol{L}$ *has finite model property*

2. $\boldsymbol{L1}$ *has finite model property*

3. $\boldsymbol{FL}$ *has finite model property*

4. $\boldsymbol{FL}_\bot$ *has finite model property*

For the first and second claim, consider [11]; the third and forth has been established by **(author?)** [22]. Theorem 2 is crucial to show that completeness for syntactic concept lattices and their reducts also holds if we restrict ourselves to languages over finite alphabets.

Theorem 3 *1.* $\mathsf{Rel}(RS) \models \Gamma \vdash \alpha$ *iff* $\Vdash_{\boldsymbol{L}} \Gamma \vdash \alpha$

2. $\mathsf{Rel}(RM) \models \Gamma \vdash \alpha$ *iff* $\Vdash_{\boldsymbol{L1}} \Gamma \vdash \alpha$

These are the canonical completeness results most important to us, established by **(author?)** [24] and **(author?)** [2]. We will provide alternative proofs for these results, because these allow us to strengthen them to relation models over finite relations. The following results have been proved by **(author?)** [28].

Theorem 4 *1.* $SCL \models \Gamma \vdash \alpha$ *if and only if* $\Vdash_{\boldsymbol{FL}_\bot} \Gamma \vdash \alpha$

2. $SCL_{\boldsymbol{FL}} \models \Gamma \vdash \alpha$ *if and only if* $\Vdash_{\boldsymbol{FL}} \Gamma \vdash \alpha$

3. $SCL_{\boldsymbol{L1}} \models \Gamma \vdash \alpha$ *if and only if* $\Vdash_{\boldsymbol{L1}} \Gamma \vdash \alpha$

The soundness directions follow *a fortiori* from Theorem 1. Completeness is proved via faithful embeddings of arbitrary bounded residuated lattices into syntactic concept lattices (which, however, yield non-standard interpretations). This embedding thus preserves falsehood, and completeness follows by contraposition. Note that this works for all logics for which $\mathbf{FL}_\bot$ is a conservative extension (i.e. same valid sequents with the connectives present), and thus for all above-mentioned, except for $\mathbf{L}$, as $\mathbf{L1}$ has more valid sequents over $/, \backslash, \bullet$ than $\mathbf{L}$ (for example $\alpha \vdash \alpha \bullet (\beta/\beta)$).

We now present a (slightly modified) reproduction of the original proof of Theorem 4, because the proof has some important features to which we will refer later on; moreover, we derive some new corollaries.

3 A Proof for Theorem 4

Let $\mathbf{B} = (B, \cdot, \vee, \wedge, /, \backslash, 1, \top, \bot)$ be a bounded residuated lattice. We denote the partial order of $\mathbf{B}$ by $\leq_B$, equality by $=_B$. That means, for terms s, t over B, $s =_B t$ states that s, t denote the same element of B. Define $\Sigma' := \{\underline{b} : b \in B\}$, and put $\Sigma = \Sigma' \cup B$. For a string $w = b_1 ... b_n \in B^*$, by $w^\bullet$ we denote the object

$b_1 \cdot ... \cdot b_n \in B$; we put $\epsilon^\bullet = 1$. Let $SI_B^* := \{b_1...b_i : (b_1...b_i)^\bullet \leq_B 1\}$ be the set of sub-identity words of B.[2] We define a language $L_B \subseteq \Sigma^*$ as the set of strings

$$L_B := \{b_1...b_n \underline{b} w : (b_1...b_n)^\bullet \leq_{\mathbf{B}} b,\, w \in SI_B^*\}.$$

Note that we now have an ambiguity as to whether a certain $b \in B$ is a letter of Σ or an element of the lattice. We could have generally avoided this ambiguity at the price of complicating notation; but we rather avoid ambiguity in all particular statements, by using $\leq_B$ or $\leq_L$ etc. Importantly, non-atomic terms over B are *not* part of the alphabet, hence for $b, b' \in B$, for example $b \wedge b' \notin \Sigma$. However, for every term t over B, there is a $b \in \Sigma$ such that $b =_B t$. Hence we have to distinguish terms from objects they denote in $\mathbf{B}$, and we have to take care to not read the terms as syntactic objects of L_B: whereas a term t will not occur in L_B unless it is an atomic term, the element it denotes does occur in L_B for every term t.

We define a map $\gamma : \wp(B^*) \to \wp(\Sigma^*)$ by $\gamma(X) = (X)^{\triangleright\triangleleft}$, where $[-]^{\triangleright\triangleleft}$ is the syntactic concept closure with respect to L_B. To simplify notation, we will henceforth often write $\gamma(b)$ instead of $\gamma(\{b\})$. Whereas γ is not a closure operator in a strict sense (as domain and co-domain differ), it is easy to see that for all $X \subseteq B^*$, $X \subseteq \gamma(X)$,[3], and we have $\gamma(b_1)\gamma(b_2) \leq \gamma(b_1 b_2)$, where by the concatenation of two sets of strings we simply mean the concatenation of their elements. So whereas γ is not strictly speaking a nucleus, it is a *nuclear map* (for a detailed treatment of this notion, see [12]).

Given a function $f : M \to N$, $X \subseteq M$, we write $f[X] = \{f(x) : x \in X\}$. A nuclear map gives rise to what is called the **nuclear image** of $\wp(B^*)$, the lattice $(\gamma[\wp(B^*)], \circ_\gamma, \cap, \cup_\gamma, /, \backslash, \gamma(\{1\}), B^*, \gamma(\emptyset))$, which we will call $\gamma(\mathbf{B})$. From the fact that γ is nuclear, it follows that $\gamma(\mathbf{B})$ is a complete residuated lattice (see [12],p.174), where $X \cup_\gamma Y := \gamma(X \cup Y)$, $X \circ_\gamma Y := \gamma(XY)$. Furthermore, $\gamma(\mathbf{B})$ is bounded by $\gamma(\top) = B^*$[4] and $\gamma(\emptyset)$; note that $\gamma(\emptyset) \neq \gamma(\{\bot\})$, as $\bot$ does *not* occur in all contexts in L_B.

It is easy to see that $\gamma(\mathbf{B})$ can be isomorphically embedded into the syntactic concept lattice $SCL(L_B)$: in fact, $\gamma(\mathbf{B})$ is isomorphic to the fragment of $SCL(L_B)$ which consists of all concepts from strings in B^*. It is easy to see that in $SCL(L_B)$, these are closed under meet, join and concatenation; but this does not follow from general considerations and is rather a consequence of the particular distributional structure of L_B. Consequently, we have an isomorphic embedding $\mathcal{C} : \gamma(\mathbf{B}) \to SCL(L_B)$, which simply maps closed sets of strings onto themselves. What we still need is an appropriate embedding from $\mathbf{B}$ into $\gamma(\mathbf{B})$.

We define a map $h : B \to \gamma(\mathbf{B})$ (equivalently, $h : B \to \gamma[\wp(B^*)]$), where $h(b) = \{w \in \Sigma^* : w\underline{b} \in L_B\}$. This is clearly a γ-closed set (or put differently: a syntactic concept of L_B), as it equals the closed set $\{(\epsilon, \underline{b})\}^\triangleleft$. In the sequel, we will often use h with terms instead of atoms. Of course, here the same applies

[2] It is well-known that for any algebra $\mathbf{A}$, the set of its subidentity elements forms a closed subalgebra.

[3] Actually, we also have $\gamma(X) \subseteq B^*$; but still we use γ as an embedding into a larger co-domain.

[4] We assume that $\top \in B$!

as before: h is *not* defined over terms, it only maps the elements denoted by the terms. A crucial lemma is the following:

Lemma 6 *For $w \in B^*, b \in B$, the following three are equivalent:*

1. $w \in h(b)$
2. $w^\bullet \leq_B b$
3. $w \leq_L b$

Proof. 1. $\Rightarrow$ 2.: assume $w \in h(b)$. Then we have $w\underline{b} \in L_B$. By definition, $w\underline{b} \in L_B$ only if $w^\bullet \leq_B b$, so the implication follows.

3. $\Rightarrow$ 1.: Assume we have $w \leq_L b$. Then $w^\triangleright \supseteq b^\triangleright$; as $(\epsilon, \underline{b}) \in b^\triangleright$, we have $(\epsilon, \underline{b}) \in w^\triangleright$, and hence $w \in \{(\epsilon, \underline{b})\}^\triangleleft = h(b)$.

2. $\Rightarrow$ 3. Assume $w^\bullet \leq_B b$. In every residuated monoid, $b \leq b'$ entails $a \cdot b \cdot c \leq a \cdot b' \cdot c$. Now if $xby \in L_B$ and y contains an occurrence $\underline{b'}$, then $xwy \in L_B$ by the definition of L_B: each word of L_B corresponds to an inequation which holds in $\mathbf{B}$, and the inequation remains valid under the substitution of $w^\bullet$ for b. Conversely, y does not contain an occurrence of $\underline{b'}$; then still substitution of b by w yields a word in SI^*_B right of $\underline{b'}$, and hence $xwy \in L_B$. □

We will now show that h defines a proper faithful embedding of $\mathbf{B}$ into the nuclear image $\gamma(\mathbf{B})$, which is a sublattice of the syntactic concept lattice of L_B.

Proposition 7 *The map h is injective, and for each $\star \in \{\wedge, \vee, \cdot, /, \backslash\}$, we have $h(a) \star_\gamma h(b) = h(a \star b)$, where $a \star b$ denotes the unique element of B denoted by the term, and $\star_\gamma$ denotes the interpretation of $\star$ in the γ-image of $\wp(B^*)$. This means that h is an faithful embedding.*

Proof. To see that h is injective, assume $h(a) = h(b)$. Obviously, we have $a \in h(a)$, $b \in h(b)$. We thus have $a \in h(b)$, hence by Lemma 6 $a \leq_B b$; but we also have $b \in h(a)$, hence $b \leq_B a$, and so $a =_B b$.

For the second claim, we proceed by cases:

Case 1: $\star = \wedge$.

a) $h(a) \cap h(b) \supseteq h(a \wedge b)$. Assume that $c \in h(a \wedge b)$. Then by Lemma 6, $c \leq_B (a \wedge b)$. Therefore, $c \leq_B a, b$, and consequently $c \leq_L a, b$. Therefore, $c \in h(a), c \in h(b)$, and thus $c \in h(a) \cap h(b)$.

b) $h(a) \cap h(b) \subseteq h(a \wedge b)$. Assume that $c \in h(a) \cap h(b)$. Then $c \in h(a)$, $c \in h(b)$; consequently, $c \leq_B a, b$ (by Lemma 6); consequently, $c \leq_B a \wedge b$, and therefore $c \in h(a \wedge b)$.

Case 2: $\star = \vee$

a) $h(a) \cup_\gamma h(b) \supseteq h(a \vee b)$: Assume that $c \in h(a \vee b)$. Then $c \leq_L a \vee b$. We show that $a \vee b \in h(a) \cup_\gamma h(b)$: whenever $wav \in L_B$, $wbv \in L_B$, then $w(a \vee b)v \in L_B$. Consequently, $a \vee b \in \gamma(\{a, b\}) \subseteq h(a) \cup_\gamma h(b)$. As γ-closed sets are downward closed w.r.t. the order $\leq_L$, we also have $c \in h(a) \cup_\gamma h(b)$.

b) $h(a) \cup_\gamma h(b) \subseteq h(a \vee b)$: By Lemma 6, we know that for all $x \in h(a)$, $x \leq_L a$, and for all $y \in h(b)$, $y \leq_L b$. Consequently, by Lemma 6 we have $z \leq_L a \vee b$ for all $z \in h(a) \cup h(b)$, and so $h(a) \cup h(b) \subseteq h(a \vee b)$. As $h(a \vee b)$ is γ-closed, we must also have $h(a) \cup_\gamma h(b) \subseteq h(a \vee b)$ by order preservation of γ.

Case 3: $\star = \cdot$.

a) $h(a) \circ_\gamma h(b) \supseteq h(a \cdot b)$. Assume that $c \in h(a \cdot b)$. Then $c \leq_L a \cdot b$, and so $c \leq_L ab$. As $h(a) \circ_\gamma h(b) = \gamma(h(a)h(b))$, we have $ab \in h(a) \circ_\gamma h(b)$, and as γ closed sets are downward closed w.r.t. $\leq_L$, we have $c \in h(a) \circ_\gamma h(b)$.

b) $h(a) \circ_\gamma h(b) \subseteq h(a \cdot b)$. We know that for all $x \in h(a), x \leq_B a$, and for all $y \in h(b), y \leq_B b$ (Lemma 6). Consequently, as it holds in any residuated lattice that $w \leq y, x \leq z$ entails $wx \leq yz$, we know that for all $x \in h(a), y \in h(b), x \cdot y \leq_B a \cdot b$, and so $xy \leq_L ab \sim_L a \cdot b$. Consequently, we have $h(a)h(b) \subseteq h(a \cdot b)$. As $h(a \cdot b)$ is closed under γ and γ preserves the inclusion order of sets, it follows that $h(a) \circ_\gamma h(b) \subseteq h(a \cdot b)$.

Case 4: $\star = /$

a) $h(a)/h(b) \supseteq h(a/b)$. Assume $c \in h(a/b)$; then $c \leq_L a/b$. We show that $a/b \in h(a)/h(b)$. By definition, $h(a)/h(b)$ is the largest element, such that $(h(a)/h(b)) \circ_\gamma h(b) \leq_{\gamma(\mathbf{B})} h(a)$. As for all $d \in h(b)$, $d \leq_L b$, and $a/b \cdot b \leq_L a$, $a \in h(a)$, we have for all $d \in h(b)$, $a/b \cdot d \leq_L a$, and therefore $a/b \cdot d \in h(a)$. It follows that $a/b \in h(a)/h(b)$, and consequently $c \in h(a)/h(b)$.

b) $h(a)/h(b) \subseteq h(a/b)$. Contraposition: assume $x \notin h(a/b)$, hence $x \not\leq_B a/b$ (Lemma 6). It follows that $xb\underline{a} \notin L_B$, as otherwise $x \cdot b \leq_B a$, and hence $x \leq_B a/b$, contradicting the assumption. It follows that $xb \notin \{(\epsilon, \underline{a})\}^\triangleleft = h(a)$, hence $xh(b) \not\subseteq h(a)$, so $x \notin h(a)/h(b)$.

Case 5: $\star = \backslash$ is parallel to 4. □

Note that the order $\leq$ is definable by $\wedge$ (or $\vee$) in every lattice, hence this result already entails that h preserves $\leq_B$. We still need to take care of the constants:

Lemma 8 *1.* $h(\top) = B^*$*, 2.* $h(\bot) = \gamma(\bot)$.

Proof. 1.We have $w\underline{\top} \in L_B$ iff $w \in B^*$, so $h(\top) = B^*$.

2. $w\underline{\bot}v \in L_B$ iff $w^\bullet \leq_B \bot$ iff $\bot =_B w^\bullet$ (as $\bot$ is $\leq_B$-minimal), and so $h(\bot) = \gamma(\bot)$. □

There is one more lemma which is quite important:

Lemma 9 $h(1) = \gamma(\epsilon)$.

Proof. $\supseteq$ We have $h(1) = SI_B^*$ by Lemma 6. Now assume $x \in \gamma(\epsilon)$. This means that we can insert it at any point in the language preserving membership, in particular to the right of $\underline{b}$ in a word $w\underline{b}$. Consequently, $x \in SI_B^*$, and hence $x \in h(1)$.

$\subseteq$ Conversely assume $x \in h(1) = SI_B^*$. In general, we have $wv\underline{b}uz \in L_B$ iff $w1v\underline{b}uz$ iff $wv\underline{b}u1z$ by Lemma 6, as $(wv)^\bullet =_B (w1v)^\bullet$; and as $x^\bullet \leq_B 1$, if

$w1v\underline{b}uz \in L$, then $wxv\underline{b}uz \in L$, and if $wv\underline{b}u1z \in L$, then $wv\underline{b}uxz \in L$. Hence $x \leq_L \epsilon$, and thus $x \in \gamma(\epsilon)$. So $h(1) \subseteq \gamma(\epsilon)$. □

This is useful not for the current embedding, but rather for the embedding of $\gamma(\mathbf{B})$ into $SCL(L_B)$. Note that this breaks down if we disallow SI_B^* suffixes, because we cannot substitute a final ϵ by 1. Moreover, note that $h(\bot)$ is not the $\bot$-element of $\gamma(\mathbf{B})$, as there are places where $\bot$ cannot figure in L_B (we can never substitute $\underline{b}$ with $\bot$). Note also that whereas $\gamma(\top)$ is the maximal element of $\gamma(\mathbf{B})$, it is *not* maximal in $SCL(L_B)$, as $\gamma(\top) = B^* \subsetneq \Sigma^*$.

We now return to the **proof of Theorem** 4. Assume that $\nvdash_{FL_\bot} \Gamma \vdash \alpha$. Then by completeness and finite model property of $\mathbf{FL}_\bot$ for bounded residuated lattices, there is a finite bounded residuated lattice $\mathbf{B}$ and assignment σ, such that $\mathbf{B}, \sigma \not\models \Gamma \vdash \alpha$, that is, in $\mathbf{B}$ we have $\overline{\sigma}(\Gamma) \not\leq \overline{\sigma}(\alpha)$.

We now take the γ-image $\gamma(\mathbf{B})$ and the embedding h, as we have defined them above. The above results ensure that for every (bounded) residuated lattice $\mathbf{B}$, s, t terms over B, and $\gamma(\mathbf{B})$, h as defined above, we have $s \leq_B t$ if and only if $h(s) \subseteq h(t)$, because h is an faithful embedding and $\subseteq, \leq_B$ are definable over the operators $\wedge, \vee$.

To complete the proof of Theorem 4, there is one step missing, which is quite straightforward: define L_B for $\mathbf{B}$ as above. Every residuated lattice $\mathbf{B}$ can be embedded into the γ-image $\gamma(\mathbf{B})$ by a map h, such that if $s \not\leq t$ in $\mathbf{B}$, then $h(s) \not\leq h(t)$ in $\gamma(\mathbf{B})$. We now have an faithful embedding $\mathcal{C} : \gamma(\mathbf{B}) \to SCL(L_B)$ of $\gamma(\mathbf{B})$ into the syntactic concept lattice of L_B, which maps the γ-closed sets onto their concepts. As the former are already closed and concepts for us are closed sets, we have $\mathcal{C}(M) = M$, that is, $\mathcal{C}$ embeds by computing the identity. We thus have the implication: if $s \not\leq t$ in $\mathbf{B}$, then there is a language L_B, such that $\mathcal{C} \circ h(s) \not\subseteq \mathcal{C} \circ h(t)$, which is to say $\mathcal{C} \circ h(s) \not\leq_{SCL(L_B)} \mathcal{C} \circ h(t)$. Consequently, if $\nvdash_{FL_\bot} \Gamma \vdash \alpha$, then there is a language L_B over a finite alphabet, such that $SCL(L_B), \mathcal{C} \circ h \circ \sigma \not\models \Gamma \vdash \alpha$. This completes the proof of Theorem 4.1. Note however that the interpretation $\mathcal{C} \circ h \circ \sigma$ is **non-standard**: we have $\mathcal{C} \circ h \circ \sigma(\top) = B^* \neq \Sigma^*$, and $\mathcal{C} \circ h \circ \sigma(\bot) = \{\bot\}^{\triangleright\triangleleft} \neq \emptyset^{\triangleright\triangleleft}$. The logical 1 is however properly interpreted, as is ensured by Lemma 9.

An important feature of our proof is the following: let the logic $\mathcal{L}$ be a fragment of $\mathbf{FL}_\bot$, such that $\mathbf{FL}_\bot$ is a conservative extension of $\mathcal{L}$. We know that these fragments exist, as $\mathbf{FL}$ is a conservative extension of $\mathbf{L1}$ and $\mathbf{L1}(\vee)$, and $\mathbf{FL}_\bot$ is a conservative extension of $\mathbf{FL}$. $\mathbf{L}$ (and its fragments) do not satisfy this requirement and pose some difficulties which we will address separately.

The algebraic notion corresponding to the notion of a fragment in logic is the notion of a reduct. So let RED be a certain class of reducts of $RL_\bot$, such that for $\mathcal{L}$ a fragment of $\mathbf{FL}_\bot$, $\mathcal{L}$ is complete with respect to RED. Then our proof of completeness regarding the class SCL of syntactic concept lattices can be easily adapted to give a proof of the completeness of a class of reducts of SCL with respect to $\mathcal{L}$, which corresponds to RED. The reason is that the crucial step, which is the embedding in Proposition 7, is equally valid for any subset of the operators $\{\vee, \wedge, \cdot, /, \backslash\}$ and constants $\{\top, \bot, 1\}$. So the question whether a reduct of SCL is complete with respect to a fragment $\mathcal{L}$ of $\mathbf{FL}_\bot$ reduces to the question whether there is a complete algebraic semantics for $\mathcal{L}$ in the sense we

have specified above. As we do have these algebraic semantics, it follows that we have a proof for the other parts of Theorem 4.

Finally, we have to consider **L**. For **L**, the above considerations are not sufficient because **L1** etc. are not conservative extensions of **L**, that is, even in the language of **L** and excluding sequents with empty antecedents, the logic **L1** has strictly more valid sequents. Recall that by $SCL^+_{\mathbf{L}}$ we denote the class of all $\circ, /, \setminus$ reducts of positive lattices $SCL^+(L)$, for any language L.

Theorem 5 *$SCL^+_{\boldsymbol{L}} \models \Gamma \vdash \alpha$ if and only if $\Vdash_{\boldsymbol{L}} \Gamma \vdash \alpha$.*

Note that these structures do not have a 1, and this is in line with the fact that in Theorem 5 we must have $\Gamma \neq \epsilon$, as **L** does not allow for empty antecedents.

Proof. A full proof of this theorem would have to reproduce most of the proof for Theorem 4. Most of the steps can be taken over literally, so we just quickly comment on the steps which are critical.

Soundness is clear. For completeness, we need to find, for an arbitrary residuated semigroup $\mathbf{S} = (S, \cdot, /, \setminus, \leq)$, a language L_S such that **S** can be isomorphically embedded into $SCL^+_{\mathbf{L}}(L_S)$. We define $L_S := \{s_1 ... s_i \underline{s} : s_1 \cdot ... \cdot s_i \leq_S s\}$ (the subidentity elements do not generally exist in this case). We can construct the γ-image of S^* as before, obtaining $\gamma(\mathbf{S})$. Moreover, it is easy to see that the analogue of Lemma 6 holds in this case (all arguments in the proof remain valid), and consequently, the embedding of Proposition 7 (suitably reduced to connectives $\circ, /, \setminus$) will work as before. Finally, we do not need to preserve any constants, in particular, there is no 1 to be preserved.

As **L** does not allow for empty sequents, there is no place where ϵ arises, so the interpretation of an **L**-sequent $\Gamma \vdash \alpha$ in S can be embedded into the γ-image $\gamma(\mathbf{S})$ over Σ^+, which in turn can be easily embedded into the positive syntactic concept lattice $SCL^+_{\mathbf{L}}(L_S)$ by simply computing the identity. □

4 Regular Languages and $SCL(REG)$

By REG we denote the class of regular languages. Given an equivalence relation $\sim$ over Σ^*, we put $[w]_\sim = \{v : w \sim v\}$, and $\Sigma^*_\sim = \{[w]_\sim : w \in \Sigma^*\}$. So $\Sigma^*_{\sim_L}$ denotes the set of $\sim_L$-congruence classes over Σ^*. We now present some very basic, yet important results for regular languages and syntactic concept lattices. Recall the following basic result:

Lemma 10 *A language $L \subseteq \Sigma^*$ is regular if and only if $\Sigma^*_{\sim_L} = \{[w]_{\sim_L} : w \in \Sigma^*\}$ is finite.*

The next is also well-known (and straightforward to prove):

Lemma 11 *$SCL(L)$ is finite if and only if L is regular.*

This is easy to see, if we keep in mind that $\sim_L$-equivalent strings are indistinguishable by means of $[-]^{\triangleright}$; hence there is only a finite number of strings over which distinct concepts can be constructed. Lemma 10 has also a straightforward application to finite algebras. Consider the following: let $\mathbf{B}$ be an arbitrary algebra equipped with a semigroup operation and a partial order respecting it, so we can define L_B as above (this covers all algebras we consider in this article). Then for $w, v \in B^*$, we have $w \sim_L v$ iff $w^\bullet =_{\mathbf{B}} v^\bullet$ (Lemma 6 is equally valid in this more general case). But recall that $B \subsetneq \Sigma$ in this case!

Lemma 12 *$\boldsymbol{B}$ is a finite algebra if and only if L_B is a regular language.*

Proof. $\Leftarrow$. Contraposition: if $\mathbf{B}$ is infinite, there is an infinite sequence of $=_B$-distinct objects $(w_1)^\bullet, (w_2)^\bullet, ... \in B$, so there are $w_1, w_2, ...$ which are not $\sim_L$-equivalent (Lemma 6).

$\Rightarrow$. We construct $L'_B = \bigcup_{b \in B} \{w\underline{b} : w^\bullet \leq_B b\}$. Assume a language $L = \{w\underline{b} : w^\bullet \leq_B b\}$ is not regular. Then $\Sigma^*_{\sim_L}$ is infinite, and there is an infinite sequence of words $w_1, w_2, ...$, such that if $i \neq j$, then $w_i \not\sim_L w_j$. So there is an infinite sequence of objects $(w_1)^\bullet, (w_2)^\bullet, ... \in B$, such that if $i \neq j$, then $(w_i)^\bullet \neq_{\mathbf{B}} (w_j)^\bullet$. Thus $\mathbf{B}$ is infinite – contradiction. Hence $\{w\underline{b} : w^\bullet \leq_B b\}$ is regular, and as B is finite, L'_B is a finite union of regular languages, which is still regular. Finally, SI^*_B is regular for the same reason as above, and so $L_B = L'_B \cdot SI^*_B$ is also regular. □

Let $\mathfrak{C}$ be a class of languages; then by $SCL(\mathfrak{C})$ we denote the class of structures $SCL(L) : L \in \mathfrak{C}$. So let REG denote the regular languages, then $SCL(REG)$ equals the class of finite syntactic concept lattices. As we have said, a finite algebra $\mathbf{B}$ entails a language L_B over a finite alphabet; the last lemma shows us that it also entails that L_B is regular. Moreover, as $\mathbf{L}, \mathbf{L1}, \mathbf{FL}, \mathbf{FL}_\bot$ have the finite model property, for completeness it is sufficient to consider only finite algebras, and consequently we can strengthen Theorem 4 to the following:

Corollary 13

1. *$SCL(REG) \models \Gamma \vdash \alpha$ if and only if $\Vdash_{\boldsymbol{FL}_\bot} \Gamma \vdash \alpha$.*
2. *$SCL_{\boldsymbol{FL}}(REG) \models \Gamma \vdash \alpha$ if and only if $\Vdash_{\boldsymbol{FL}} \Gamma \vdash \alpha$.*
3. *$SCL_{\boldsymbol{L1}}(REG) \models \Gamma \vdash \alpha$ if and only if $\Vdash_{\boldsymbol{L1}} \Gamma \vdash \alpha$.*
4. *$SCL^+_{\boldsymbol{L}}(REG) \models \Gamma \vdash \alpha$ if and only if $\Vdash_{\boldsymbol{L}} \Gamma \vdash \alpha$.*

5 Automata-theoretic Semantics

5.1 Automata-theoretic Preliminaries

We now introduce a new class of bounded residuated lattices, the **automatic concept lattices**. It is very similar to SCL in that it is based on a Galois connection which, provided the certain conditions, gives rise to a nucleus. As we will learn from the main result of this section, the Isomorphism Theorem, if we consider structures only up to isomorphism, then automatic concept lattices

form a proper generalization of syntactic concept lattices (in fact, in general they are not even residuated lattices).

One can present automata in many different ways, the most standard one being probably the following: an automaton as *state-transition system* is a tuple $\mathfrak{A} = (\Sigma, Q, \delta, F, I)$, where Σ is a finite input alphabet, Q a set of states, $\delta \subseteq Q \times \Sigma \times Q$ a transition relation, $F \subseteq Q$ a set of accepting states, $I \subseteq Q$ the set of initial states. This notation of automata is somewhat clumsy in connection with the techniques we use later on, so we will choose a slightly different presentation which we call **relational**. This is a notional change we adopt for convenience, where the transformation is easy to perform and does not come with any substantial changes. We define a **semi-automaton** as a tuple $\langle \Sigma, \phi \rangle$, where ϕ is a map $\phi : \Sigma \to \wp(Q \times Q)$, mapping letters in Σ onto relations over Q, where we use Q as an arbitrary (finite or infinite) carrier set. A transition $(q, a, q') \in \delta$ then corresponds to $(q, q') \in \phi(a)$ (and vice versa). ϕ is extended to strings by interpreting concatenation as **relation composition** '**;**', where $R;R' = \{(x, y) : (x, z) \in R, (z, y) \in R'\}$. So we have $\phi(aw) = \phi(a);\phi(w)$, and ϕ is a homomorphism from the free monoid Σ^* into a relation monoid over Q, and a word $w \in \Sigma^*$ then induces a relation $\phi(w) \subseteq Q \times Q$. Defining ϕ as a homomorphism, we should take care of $\phi(\epsilon)$, which we simply define by $\phi(\epsilon) = \mathsf{id}_Q := \{(q, q) : q \in Q\}$.[5] To get a full automaton, we still need an *accepting relation*. One usually specifies a set of initial and accepting states, yielding an accepting relation $I \times F$. As for us, acceptance will only play a minor role, we will take a slightly more general convention and assume that automata specify an **accepting relation** $F_R \subseteq Q \times Q$. Thus a full **automaton** is a tuple $\langle \Sigma, \phi, F_R \rangle$. We define the language recognized by an automaton $\mathcal{A} = \langle \Sigma, \phi, F_R \rangle$ by $L(\mathcal{A}) := \{w \in \Sigma^* : \phi(w) \cap F_R \neq \emptyset\}$.

5.2 Automatic Concepts

In what is to follow, we will take the "canonical view" on formal concepts, that is: concepts are not simply $[-]^{\triangleright\triangleleft}$-closed sets, but pairs (M, C) such that $M^{\triangleright} = C$, $C^{\triangleleft} = M$ (this entails that both are closed). Henceforth, we will use the maps $[-]^{\triangleright}, [-]^{\triangleleft}$ for syntactic concepts only. Given a semi-automaton $\langle \Sigma, \phi \rangle$, $M \subseteq \Sigma^*$, $R \subseteq Q \times Q$, we define the two polar maps

$$M^{\blacktriangleright} = \bigcap_{w \in M} \phi(w) \tag{3}$$

$$R^{\blacktriangleleft} = \{w : \phi(w) \supseteq R\} \tag{4}$$

[5] But in principle, nothing prevents us from having $(x, y) \in \phi(\epsilon)$ with $x \neq y$ – we just have to make sure that for all $a \in \Sigma$, we have $\phi(\epsilon);\phi(a) = \phi(a) = \phi(a);\phi(\epsilon)$. One might also think this gives rise to another problem, namely when we have automata with ϵ transitions. However, ϵ-transitions in a (classical) automaton have to be distinguished from $\phi(\epsilon)$, which is algebraic in nature: assume we have an ϵ-transition R_ϵ, in addition to transitions $R_a : a \in \Sigma$. In the algebraic setting, this is part of primitive transitions: we have to define $\phi(a)$ as the smallest set such that 1. $R_a \subseteq \phi(a)$, and 2. $\phi(a);R_\epsilon \subseteq \phi(a)$, $R_\epsilon;\phi(a) \subseteq \phi(a)$.

To get an intuition, imagine a standard automaton with states and transitions between them. $\{w\}^{\blacktriangleright}$ is then the set of all (q, q') such that we can reach q' from q while reading w; $\{w, v\}^{\blacktriangleright}$ is the set of all (q, q') such that we can reach q' from q both by reading w and v, and so on. In the same way, $\{(q, q')\}^{\blacktriangleleft}$ is the set of all words w such that we can reach q' from q while reading w.

It is easy to see that $[-]^{\blacktriangleright}, [-]^{\blacktriangleleft}$ establish a Galois connection and their compositions $[-]^{\blacktriangleright\blacktriangleleft}$, $[-]^{\blacktriangleleft\blacktriangleright}$ are closure operators. An **automatic concept** is then a pair (M, R) with $M^{\blacktriangleright} = R$, $R^{\blacktriangleleft} = M$ (of course, the underlying (semi-)automaton is understood as given). We denote the set of automatic concepts, given an automaton $\mathcal{A}$, by $\mathfrak{A}_{\mathcal{A}}$. Importantly, the map $[-]^{\blacktriangleright\blacktriangleleft}$ does **not** form a nucleus on Σ^*, and in general, $[-]^{\blacktriangleright\blacktriangleleft}$-closed concatenation does not distribute over infinite joins. Consequently, we cannot simply define a residuated lattice of concepts in the usual fashion. Rather, we have to restrict our attention to a certain class of automata.

Definition 14 *A (semi-)automaton $\langle \Sigma, \phi, F_R) \rangle$ is **nuclear**, if for all $M, N \subseteq \Sigma^*$, $(\bigcap_{w \in M} \phi(w)) ; (\bigcap_{v \in N} \phi(w)) = \bigcap_{wv \in MN} \phi(wv)$.*

Note that $\subseteq$ always holds. The equality ensures that $[-]^{\blacktriangleright\blacktriangleleft}$ is a nucleus on Σ^*, because if $w \in M^{\blacktriangleright\blacktriangleleft}$, $v \in N^{\blacktriangleright\blacktriangleleft}$, then $\phi(w) \supseteq M^{\blacktriangleright}, \phi(v) \supseteq N^{\blacktriangleright}$. Hence $\phi(wv) = \phi(w) ; \phi(v) \supseteq M^{\blacktriangleright} ; N^{\blacktriangleright} = (MN)^{\blacktriangleright}$ (by Definition 14), and hence $wv \in (MN)^{\blacktriangleright\blacktriangleleft}$, thus entailing $M^{\blacktriangleright\blacktriangleleft} N^{\blacktriangleright\blacktriangleleft} \subseteq (MN)^{\blacktriangleright\blacktriangleleft}$. So being nuclear boils down to composition distributing over (infinite) intersections of closed sets. We will later see that for every automaton there is a nuclear automaton recognizing the same language. We define

$$(M, R) \wedge (N, S) = (M \cap N, (R \cup S)^{\blacktriangleleft\blacktriangleright}) \tag{5}$$

$$(M, R) \vee (N, S) = ((M \cup N)^{\blacktriangleright\blacktriangleleft}, R \cap S) \tag{6}$$

$$(M, R) \circ (N, S) = ((MN)^{\blacktriangleright\blacktriangleleft}, (MN)^{\blacktriangleright}) \tag{7}$$

It is easy to see that $\wedge, \vee$ can be extended to the infinitary operators $\bigwedge, \bigvee$ (as they are based on sets). Moreover, in case the underlying automaton is nuclear, $\circ$ distributes over infinite joins (because it is a nuclear operation), so the residuals are easily defined in the usual fashion by

$$M/N = \bigvee \{X : X \circ N \leq M\},\ N \backslash M = \bigvee \{X : N \circ X \leq M\}.$$

We put $\top = (\Sigma^*, (\Sigma^*)^{\blacktriangleright})$, $\bot = (\emptyset^{\blacktriangleright\blacktriangleleft}, \emptyset^{\blacktriangleright})$, where by convention we put $\emptyset^{\blacktriangleright} = \bigcup_{w \in \Sigma^*} \phi(w)$. Finally, we put $1 = (\{\epsilon\}^{\blacktriangleright\blacktriangleleft}, \phi(\epsilon))$ (recall that $\phi(\epsilon) = \mathsf{id}_Q$ by definition). So given a nuclear automaton $\mathcal{A}$, we have the complete bounded residuated lattice $(\mathfrak{A}_{\mathcal{A}}, \circ, \wedge, \vee, /, \backslash, 1, \top, \bot)$, which is the **automatic concept lattice** of $\mathcal{A}$, for short $ACL(\mathcal{A})$. As is easy to see, acceptance does not play a role for the automatic concept lattice, so it is sufficient to refer to semi-automata. By ACL we denote the class of all $ACL(\mathcal{A})$ for $\mathcal{A}$ an arbitrary nuclear (semi-)automaton, and we define the reducts $ACL_{\mathbf{FL}}, ACL_{\mathbf{L1}}$ in the same way we did for SCL.

We will refer to the straightforward interpretation of $\mathbf{FL}_\bot$ and its fragments into automatic concept lattices as **automata-theoretic semantics**, and write $ACL \models \Gamma \vdash \alpha$ in the usual sense that for all nuclear semi-automata $\mathcal{A}$, interpretations σ into $ACL(\mathcal{A})$, we have $\overline{\sigma}(\Gamma) \leq_{ACL(\mathcal{A})} \overline{\sigma}(\alpha)$; same for reducts $ACL_{\mathbf{FL}}, ACL_{\mathbf{L1}}$ etc.

For $ACL(\langle \phi, \Sigma, F_R \rangle)$, F_R is irrelevant. Still, F_R is useful because it links automata to languages, which in turn is necessary to establish the relation between ACL and SCL. For what is to follow, the phrase "automaton recognizing L" could be exchanged with "semi-automaton $\langle \phi, \Sigma \rangle$ for which there is F_R such that $L(\langle \phi, \Sigma, F_R \rangle) = L$", which however is clumsy to repeat. As automata are related to languages, there should be thus a relation between $ACL(\mathcal{A})$ and $SCL(L)$, provided that $L(\mathcal{A}) = L$. In particular, one knows that in this case, if $w \not\sim_L v$, then $\phi(w) \neq \phi(v)$ – otherwise, the automaton could not distinguish acceptance of words containing the two substrings. The inverse direction is obviously incorrect, that is, $\phi(w) \neq \phi(v)$ does not imply anything for w, v in L, as the automaton can make as many (unnecessary) distinctions as it desires (this is related to the issue of minimality of automata). From this, we can for example conclude the following: if $L(\mathcal{A}) = L$, then for $(M, C) \in \mathcal{B}_L$, there are $(M_i, R_i) \in \mathfrak{A}_\mathcal{A}$ for $i \in I$, such that $M = \bigcup_{i \in I} M_i$. However, this does **not** entail (as one might conjecture) that we have $M = (\bigcup_{i \in I} M_i)^{\blacktriangleright\blacktriangleleft}$, which by completeness of the lattice would entail that there is an automatic concept $(M, R) \in \mathfrak{A}_\mathcal{A}$. For example, imagine the following situation: for every $w \in M$, there is $(r_w, r'_w) \in \phi(m)$, such that $\phi(x) \circ \{(r_w, r'_w)\} \circ \phi(y) \cap F \neq \emptyset$, but $(r_w, r'_w) \notin M^{\blacktriangleright}$. So every $w \in M$ has its own peculiar pair which makes sure $xwy \in L$. Obviously, for $\bigvee_{w \in M} (\{w\}^{\blacktriangleright\blacktriangleleft}, \phi(w)) = (N, R)$, we have $N \supseteq M$. Still there can be $v \in N$, $v \notin M$, because $\phi(v) \supseteq R$, but $\phi(v)$ does not contain *any* of the pairs which ensure that $xMy \subseteq L$, and in fact $xvy \notin L$.

So in general, there is no homomorphic relation between the two structures, so there is no trivial way to extend completeness for SCL to completeness for automata-theoretic semantics via embeddings; instead, we have to recur to a peculiar automata-theoretic construction.

5.3 The Universal Automaton

There are always infinitely many distinct automata recognizing a language (even modulo a labelled-graph based notion of automaton-isomorphism). We will now consider a particular automaton type which is uniquely specified for every language and which allows us to connect syntactic concepts to automatic concepts. This is the so-called **universal automaton** [see 19]. The idea that there is a connection between syntactic concepts and the universal automaton is goes back to A. Clark (see [9], where, among other, the relation between factorizations and $[-]^{\triangleright\triangleleft}$-closed sets is investigated). However, the direct correlation we establish here is new to my knowledge. The universal automaton is based on the notion of a factorization of a language. (X, Y) is a **factorization** of L, iff

1. $XY \subseteq L$, and

2. if $X \subseteq X', Y \subseteq Y'$ and $X'Y' \subseteq L$, then $X = X', Y = Y'$.

We denote the set of L-factorizations with $fact(L)$. So a factorization is a maximal decomposition of L into two factors. We denote the (unique) universal automaton for a language L by $U(L)$. The factorizations of L form the set of states of $U(L)$. We define I, the set of initial factorizations and F, the set of final factorizations as follows: $I = \{(X, Y) \in fact(L) : \epsilon \in X\}$, $F = \{(X, Y) \in fact(L) : \epsilon \in Y\}$. Then for $L \subseteq \Sigma^*$, one defines the **universal automaton** $U(L) := (\Sigma, fact(L), I, F, \delta)$, where for $a \in \Sigma$, $((X, Y), a, (X', Y')) \in \delta$ iff $Xa \subseteq X'$ iff $Y \supseteq aY'$. The latter bi-implication is easy to see: if $Xa \subseteq X'$, then $XaY' \subseteq L$, and so $aY' \subseteq Y$ (same for the other direction). All results of this subsection can be found in [19]; we present them as they are necessary for the proof of the Isomorphism Theorem, but we omit the proofs. Until now, we have given the "normal" presentation of universal automata. To proceed, we quickly need to bring the universal automaton into our "relational normal form" for automata: we put $U(L) = \langle \Sigma, \phi, I \times F \rangle$, where for all $a \in \Sigma$, we have

$$\phi(a) = \{((X, Y), (X', Y')) : Xa \subseteq X' \text{ and } (X, Y), (X', Y') \in fact(L)\} \tag{8}$$

Factorizations give also rise to a Galois connection and closure operators in a natural way [see 9]; we put

$$M^{\rightarrow} = \{w : Mw \subseteq L\} \tag{9}$$

$$M^{\leftarrow} = \{w : wM \subseteq L\} \tag{10}$$

The compositions $[-]^{\rightarrow\leftarrow}, [-]^{\leftarrow\rightarrow}$ are closure operators, and $[-]^{\rightarrow}, [-]^{\leftarrow}$ establish a Galois connection between closed sets of strings. A factorization is then exactly a pair of sets (M, N) such that $M^{\rightarrow} = N$, $N^{\leftarrow} = M$ (this entails $M = M^{\rightarrow\leftarrow}, N = N^{\leftarrow\rightarrow}$). Depending on L, there might be trivial factorizations $(\Sigma^*, \emptyset)$, $(\emptyset, \Sigma^*)$.

Lemma 15 *For $(X, Y), (X', Y') \in fact(L)$, $W \subseteq \Sigma^*$, the following are equivalent:*

1. $XW \subseteq X'$
2. $WY' \subseteq Y$
3. $XWY' \subseteq L$.

Lemma 16 *For all $w \in \Sigma^*$, $((X, Y), (X', Y')) \in \phi(w)$ iff $Xw \subseteq X'$ iff $wY' \subseteq Y$ iff $XwY' \subseteq L$.*

Lemma 17 *$L(U(L)) = L$.*

That is, the universal automaton of L recognizes L. It is a straightforward consequence of the Myhill-Nerode theorem that $fact(L)$ is finite if and only if L is regular. It entails the following:

Lemma 18 *$U(L)$ is a finite automaton if and only if L is regular.*

5.4 Completeness and an Isomorphism Theorem for ACL and SCL

We have said that there is no homomorphic (or in fact, any simple structural) relation between $SCL(L)$ and $ACL(\mathcal{A})$ for all $\mathcal{A}$ such that $L(\mathcal{A}) = L$. This is despite the fact that $\mathcal{A}$ must make the relevant distinctions between strings distinct modulo $\sim_L$. One problem is the following:

Lemma 19 *In general,* $(MN)^{\blacktriangleright} \supsetneq M^{\blacktriangleright};N^{\blacktriangleright}$.

Proof. $\supseteq$ $(MN)^{\blacktriangleright} = \bigcap_{wv \in MN} \phi(wv) = \bigcap_{w \in M, v \in N} (\phi(w);\phi(v))$, whereas $M^{\blacktriangleright};N^{\blacktriangleright} = (\bigcap_{w \in M} \phi(w));\bigcap_{v \in N} \phi(v))$. In general, $\cap$ does not distribute over $;$. However, assume that $(x,y) \in M^{\blacktriangleright};N^{\blacktriangleright}$. Then there is a z such that $(x,z) \in M^{\blacktriangleright}$, $(z,y) \in N^{\blacktriangleright}$. Hence we have for all $w \in M, v \in N$, $(x,z) \in \phi(w), (z,y) \in \phi(v)$, and hence for all $w \in M, v \in N$, $(x,y) \in \phi(w);\phi(v) = \phi(wv)$; hence $(x,y) \in (MN)^{\blacktriangleright}$.

$\neq$ The underlying reason $M^{\blacktriangleright};N^{\blacktriangleright}$ is generally smaller than $(MN)^{\blacktriangleright}$ is that $(x,y) \in (MN)^{\blacktriangleright}$ iff for all $w \in M, v \in N$, we have *some* z (not necessarily the same) such that $(x,z) \in \phi(w), (z,y) \in \phi(v)$, whereas in $M^{\blacktriangleright};N^{\blacktriangleright}$, the z such that $(x,z) \in \phi(w), (z,y) \in \phi(v)$ must be the same for all $w \in M, v \in N$. We give a concrete example. Take $M = \{a,b\}, N = \{c,d\}$, $\phi(a) = \{(e,f)\}$, $\phi(b) = \{(e,g)\}$, $\phi(c) = \{(f,e),(g,e)\}$, $\phi(d) = \{(f,e),(g,e)\}$. Then $(MN)^{\blacktriangleright} = \bigcap\{\phi(w) : w \in MN\}$, and thus $(e,e) \in (MN)^{\blacktriangleright}$. Conversely, $M^{\blacktriangleright} = \emptyset$, and so $M^{\blacktriangleright};N^{\blacktriangleright} = \emptyset$. □

Note that even closing the right-hand side under $[-]^{\blacktriangleleft\blacktriangleright}$ does not help. Things change however if we look at the universal automaton instead of automata in general, as we will see in the Isomorphism Theorem. The following generalization of Lemma 16 is quite simple, but will be very helpful. Let $[-]^{\blacktriangleright}, [-]^{\blacktriangleleft}$ below be defined with respect to $U(L)$.

Lemma 20 *For* $(X,Y),(X',Y') \in \mathit{fact}(L)$, $((X,Y),(X',Y')) \in M^{\blacktriangleright}$ *if and only if* $XMY' \subseteq L$.

Proof. *If*: Assume $XMY' \subseteq L$. Then for every $w \in M$, we have $XwY' \subseteq L$, hence $((X,Y),(X',Y')) \in \phi(w)$, hence $((X,Y),(X',Y')) \in M^{\blacktriangleright}$.

Only if: Assume $((X,Y),(X',Y')) \in M^{\blacktriangleright}$. Then for all $w \in M$, we have $((X,Y),(X',Y')) \in \phi(w)$. Hence for all $w \in M$, $XwY' \subseteq L$, so $XMY' \subseteq L$. □

We can now show that universal automata are nuclear, so they provide a sound semantics for the full Lambek calculus.

Lemma 21 *Let* $[-]^{\blacktriangleright}, [-]^{\blacktriangleleft}$ *be defined with respect to* $U(L)$ *for some language* L. *Then* $M^{\blacktriangleright};N^{\blacktriangleright} = (MN)^{\blacktriangleright}$. *Hence for every language* L, $U(L)$ *is nuclear.*

Proof. $\subseteq$ Holds in general, see Lemma 19.

$\supseteq$ Assume $((X,Y'),(X',Y)) \in (MN)^{\blacktriangleright}$. Then $XMNY \subseteq L$.

Firstly, we show that $((X,Y'),((NY)^{\leftarrow},(NY)^{\leftarrow\rightarrow}) \in M^{\blacktriangleright}$: we have $(X,Y') \in \mathit{fact}(L)$ by assumption, $((NY)^{\leftarrow},(NY)^{\leftarrow\rightarrow}) \in \mathit{fact}(L)$ by definition of $[-]^{\leftarrow}$,

$[-]^{\rightarrow}$, and since $XMNY \subseteq L$, we also have $XM(NY)^{\leftarrow\rightarrow} \subseteq L$. So the claim follows from Lemma 20.

Secondly, we can show that $(((NY)^{\leftarrow}, (NY)^{\leftarrow\rightarrow}), (X', Y)) \in N^{\blacktriangleright}$: $(X', Y) \in \mathit{fact}(L)$ by assumption, and we have $((NY)^{\leftarrow}NY \subseteq L$ by definition of $[-]^{\leftarrow}$, hence the claim follows again from Lemma 20.

Consequently, by definition of ;, we have $((X, Y'), (X', Y)) \in M^{\blacktriangleright};N^{\blacktriangleright}$. □

In the sequel, $[-]^{\triangleright}, [-]^{\triangleleft}$ refer to SCL-closure w.r.t. to some fixed $L \subseteq \Sigma^*$, $[-]^{\blacktriangleright}, [-]^{\blacktriangleleft}$ to ACL-closure w.r.t. to $U(L)$ (both referring to the same language!). Now comes the crucial lemma for the Isomorphism Theorem:

Lemma 22 *For all $M \subseteq \Sigma^*$, $M^{\triangleright\triangleleft} = M^{\blacktriangleright\blacktriangleleft}$.*

Proof. $M^{\triangleright\triangleleft} \subseteq M^{\blacktriangleright\blacktriangleleft}$. Assume $w \in M^{\triangleright\triangleleft}$. Then whenever $xMy \subseteq L$, then $xwy \in L$. If $((X, Y), (X', Y')) \in M^{\blacktriangleright}$, then $XMY' \subseteq L$ (by Lemma 20). However, if $XMY' \subseteq L$, then $XwY' \subseteq L$, hence (by the equivalence in Lemma 15) $Xw \subseteq X', wY' \subseteq Y$. Hence we have $((X, Y), (X', Y')) \in \phi(w)$ for all $((X, Y), (X', Y')) \in M^{\blacktriangleright}$. Hence we have $w \in M^{\blacktriangleright\blacktriangleleft}$.

$M^{\blacktriangleright\blacktriangleleft} \subseteq M^{\triangleright\triangleleft}$. Assume $w \in M^{\blacktriangleright\blacktriangleleft}$, and take an arbitrary $(x, y) \in M^{\triangleright}$. Put $X = (My)^{\leftarrow}$, $Y = (XM)^{\rightarrow}$. It is easy to see that 1. $x \in X, y \in Y$ (by construction, as $xMy \subseteq L$), and 2. $((X, X^{\rightarrow}), (Y^{\leftarrow}, Y)) \in M^{\blacktriangleright}$ (by Lemma 20, as $XMY \subseteq L$). Since $w \in M^{\blacktriangleright\blacktriangleleft}$, we have $M^{\blacktriangleright} \subseteq \phi(w)$, and so $((X, X^{\rightarrow}), (Y^{\leftarrow}, Y)) \in \phi(w)$, which holds iff $XwY \subseteq L$, entailing $xwy \in L$. Hence, as $(x, y) \in M^{\triangleright}$ was arbitrary, $w \in M^{\triangleright\triangleleft}$. □

This already entails that the operations and constants in the respective lattices yield the same result, because they are based on the same underlying set-operations, of which we simply take the (same) closure. We denote operations in $SCL(L)$ as usual; the operation in $ACL(U(L))$ corresponding to $\star$ in $SCL(L)$ will be denoted by $\star'$. We distinguish the constants of different structures by subscripts as in $\top_{SCL(L)}$ etc. As concepts are tuples, we write, for tuples $(X_1, X_2), (Y_1, Y_2)$, $(X_1, X_2) =_1 (Y_1, Y_2)$ iff $X_1 = Y_1$, that is, if their first components are identical.

Corollary 23 *For $\star \in \{\wedge, \vee, \circ, /, \backslash\}$, $\star$ defined w.r.t. $SCL(L)$, $\star'$ defined w.r.t. $ACL(U(L))$,*

1. $(M, M^{\triangleright}) \star (N, N^{\triangleright}) =_1 (M, M^{\blacktriangleright}) \star' (N, N^{\blacktriangleright})$.
2. $\top_{SCL(L)} =_1 \top_{ACL(U(L))}$
3. $\bot_{SCL(L)} =_1 \bot_{ACL(U(L))}$
4. $1_{SCL(L)} =_1 1_{ACL(U(L))}$

As we have said, 1. is now straightforward, as all operations take the same closure of the same underlying set-theoretic operation. 2.-4. are no less clear, as the constants are simply obtained as the closure of the same objects (these arguments obviously only concern the left component of concepts).

For two algebras $\mathbf{B}, \mathbf{B}'$, we write $\mathbf{B} \cong \mathbf{B}'$ if there is an isomorphism from one to the other, that is a bijection which preserves all results of all operations. It is now easy to construct an isomorphism $i : SCL(L) \to ACL(U(L))$: for every $(M, C) \in \mathcal{B}_L$, we put $i(M, C) = (M, M^{\blacktriangleright})$. This completes the proof of Theorem 6:

Theorem 6 *(Isomorphism Theorem)* $ACL(U(L)) \cong SCL(L)$

That is, the automatic concept lattice for the universal automaton over L is isomorphic to the syntactic concept lattice of L. The Isomorphism Theorem thus establishes a surprisingly strong connection between the syntactic concept lattice and the universal automaton of a language.

We now consider the consequences of the Isomorphism Theorem for our investigations into the semantics of substructural logics. Automata-theoretic semantics is richer than simple language-theoretic semantics, because there is a many-one relationship of recognition between automata and languages. In order to ensure soundness, we already have to restrict interpretations to nuclear automata; then soundness follows from more general results. To obtain completeness, the Isomorphism Theorem can be applied in a straightforward fashion: just compose the SCL-interpretation of $\mathbf{FL}_\bot$ (or its fragments) with the isomorphism from $SCL(L)$ into $ACL(U(L))$, and we are done.

Theorem 7 *(Completeness of automata-theoretic semantics)*

1. $ACL \models \Gamma \vdash \alpha$ *iff* $\Vdash_{\boldsymbol{FL}_\bot} \Gamma \vdash \alpha$
2. $ACL_{\boldsymbol{FL}} \models \Gamma \vdash \alpha$ *iff* $\Vdash_{\boldsymbol{FL}} \Gamma \vdash \alpha$
3. $ACL_{\boldsymbol{L1}} \models \Gamma \vdash \alpha$ *iff* $\Vdash_{\boldsymbol{L1}} \Gamma \vdash \alpha$

It is obvious how to further strengthen these results: let $ACL(FIN)$ denote the class of automatic concept lattices over finite automata (i.e. automata with finite state set).[6] We can depart from completeness for $SCL(REG)$: for $\nVdash_{\mathbf{FL}_\bot} \Gamma \vdash \alpha$ we find a countermodel $SCL(L) : L \in REG$. By the Isomorphism Theorem, we also have a countermodel $ACL(U(L))$, which is finite. Thus we have the following:

Theorem 8 *(Completeness for finite automata)*

1. $ACL(FIN) \models \Gamma \vdash \alpha$ *iff* $\Vdash_{\boldsymbol{FL}_\bot} \Gamma \vdash \alpha$
2. $ACL_{\boldsymbol{FL}}(FIN) \models \Gamma \vdash \alpha$ *iff* $\Vdash_{\boldsymbol{FL}} \Gamma \vdash \alpha$
3. $ACL_{\boldsymbol{L1}}(FIN) \models \Gamma \vdash \alpha$ *iff* $\Vdash_{\boldsymbol{L1}} \Gamma \vdash \alpha$

[6] This class is strictly smaller than the class of automata recognizing regular languages, as obviously there are infinite automata recognizing regular languages.

All results can be extended to **L** as well: it is easy to see that the Isomorphism Theorem can be extended to hold for $SCL^+(L)$ and $ACL^+(U(L))$, because as the closure operators $[-]^{\triangleright\triangleleft}, [-]^{\blacktriangleright\blacktriangleleft}$ coincide, so do $[-]^{+\triangleright+\triangleleft}$ and $[-]^{+\blacktriangleright+\blacktriangleleft}$. Note that the absence of ϵ implies that in general, we need not have the identity relation in any concept, because we do not have $\phi(\epsilon)$! Thus we easily get the following:

Theorem 9 $ACL^+_{\boldsymbol{L}}(FIN) \models \Gamma \vdash \alpha$ *iff* $\Vdash_{\boldsymbol{L}} \Gamma \vdash \alpha$

6 Some General Facts on Nuclei and Residuated Lattices

We now present some rather simple results on nuclei and residuated lattices. These results are surely folklore; the reason we present them is that they establish a sort of a road map for the rest of this article, as we now have finished the part on semantics involving closure operators, and proceed with the canonical semantics. There is an underlying pattern which works for completeness proofs going from semantic involving nuclei to canonical semantics: assume γ is a nucleus on a canonical model (of languages or relations), and $\star$ is a canonical operation corresponding to the logical connective $\star_l$. The last sections have given us completely satisfying completeness results for structures using connectives of the form $\star_\gamma$ with a nuclear map as defined in Proposition 7. If we would have a general result making sure that

(11) $\gamma(a) \star_\gamma \gamma(b) = \gamma(a \star b)$

then we could easily deduce completeness for the canonical operations $\star$: If $\Gamma \nvdash \alpha$, by our completeness results so far we have $\overline{\sigma}(\Gamma) \not\leq \overline{\sigma}(\alpha)$, $\overline{\sigma}$ interpreting a logical connective $\star_l$ as $\star_\gamma$. Now take an interpretation $\overline{\tau}$, where for $x \in Pr$, $\overline{\tau}(x) = \overline{\sigma}(x)$, but where $\overline{\tau}(\alpha \star_l \beta) = \overline{\tau}(\alpha) \star \overline{\tau}(\beta)$; that is, $\overline{\tau}$ is a canonical interpretation without γ-closure, whereas $\overline{\sigma}(\alpha \star_l \beta) = \overline{\sigma}(\alpha) \star_\gamma \overline{\sigma}(\beta)$. As γ is a closure operator, it would be easy to extend completeness for $\overline{\sigma}$ to completeness for $\overline{\tau}$ by contraposition: because of (11), we would have $\gamma(\overline{\tau}(\alpha)) = \overline{\sigma}(\alpha)$, and as γ is a closure operator, $\overline{\sigma}(\alpha) \not\leq \overline{\sigma}(\beta)$ implies $\overline{\tau}(\alpha) \not\leq \overline{\tau}(\beta)$. Unfortunately, things are not that easy, because (11) does not generally hold.

Lemma 24 *Let* $\boldsymbol{B} = (B, \wedge, \vee, \cdot, /, \backslash, 1)$ *be a residuated lattice,* $\gamma : B \to B$ *a nucleus. Then for all* $b, b' \in B$*, the following hold:*

1. $\gamma(\gamma(b) \cdot \gamma(b')) = \gamma(b \cdot b')$
2. $\gamma(\gamma(b) \vee \gamma(b')) = \gamma(b \vee b')$
3. $\gamma(\gamma(b)/\gamma(b')) = \gamma(b)/\gamma(b') = \gamma(b)/b' \geq \gamma(b/b')$
4. $\gamma(\gamma(b)\backslash\gamma(b')) = \gamma(b)\backslash\gamma(b') = b\backslash\gamma(b') \geq \gamma(b\backslash b')$
5. $\gamma(b) \wedge \gamma(b) \geq \gamma(b \wedge b')$

Proof. 1. $\leq$ We have $\gamma(b \cdot b') \geq \gamma(b) \cdot \gamma(b')$ (nucleus), hence $\gamma(\gamma(b \cdot b')) \geq \gamma(\gamma(b) \cdot \gamma(b'))$ (extensiveness); but $\gamma(\gamma(b \cdot b')) = \gamma(b \cdot b')$ (idempotency).

$\geq$ We have $\gamma(b) \geq b, \gamma(b') \geq b'$, hence $b \cdot b' \leq \gamma(b) \cdot \gamma(b')$, hence $\gamma(b \cdot b') \leq \gamma(\gamma(b) \cdot \gamma(b'))$.

2. $\leq \gamma(b \vee b') \geq \gamma(b), \gamma(b')$, hence $\gamma(b \vee b') \geq \gamma(b) \vee \gamma(b')$; hence $\gamma(\gamma(b \vee b')) \geq \gamma(\gamma(b) \vee \gamma(b'))$, and $\gamma(\gamma(b \vee b')) = \gamma(b \vee b')$.

$\geq$ See case 1., which is parallel.

3. a) $\gamma(\gamma(b)/\gamma(b')) = \gamma(b)/\gamma(b')$ We have $x \leq \gamma(b)/\gamma(b')$ iff $x \cdot \gamma(b') \leq \gamma(b)$ iff $\gamma(x) \cdot \gamma(b') \leq \gamma(b)$ iff $\gamma(x) \leq \gamma(b)/\gamma(b')$, so the claim follows.

b) $\gamma(b)/\gamma(b') = \gamma(b)/b')$ We have $x \leq \gamma(b)/\gamma(b')$ iff $x \cdot \gamma(b') \leq \gamma(b)$. Obviously, this entails $x \cdot b' \leq \gamma(b)$, and so $x \leq \gamma(b)/b'$. Conversely, assume $x \leq \gamma(b)/b'$; then $x \cdot b' \leq \gamma(b)$; this entails $\gamma(x \cdot b') \leq \gamma(\gamma(b)) = \gamma(b)$. As f is a nucleus, $x \cdot \gamma(b') \leq \gamma(x) \cdot \gamma(b') \leq \gamma(x \cdot b')$, hence $x \cdot \gamma(b') \leq \gamma(b)$, hence $x \leq \gamma(b)/\gamma(b')$.

c) $\gamma(b)/b' \geq \gamma(b/b')$ Assume $x \leq \gamma(b/b')$. Then there is a y such that $y \leq b/b'$, $\gamma(y) \geq x$. We have $y \cdot b' \leq b$, and hence $\gamma(y) \cdot \gamma(b') \leq \gamma(y \cdot b') \leq \gamma(b)$, hence $x \leq \gamma(y) \leq \gamma(b)/\gamma(b')$.

4. Parallel.

5. We have $b \wedge b' \leq b, b'$, hence $\gamma(b \wedge b') \leq \gamma(b), \gamma(b')$, so $\gamma(b \wedge b') \leq \gamma(b) \wedge \gamma(b')$

□

Lemma 25 *There is some residuated lattice $\boldsymbol{B}$, a nucleus $\gamma : B \to B$, such that for some $b, b' \in B$ we have*

1. $\gamma(b)/b' > \gamma(b/b')$
2. $b' \backslash \gamma(b) > \gamma(b \backslash b')$
3. $\gamma(b) \wedge \gamma(b') > \gamma(b \wedge b')$.

We skip the proof of these rather obvious facts, as they are more tedious than difficult. Instead we ponder a minute about the meaning of these results. Cases 1,2 of Lemma 24 allow us to deduce completeness for canonical L-models immediately for these cases. The problem is that the equality does *not* hold for $\backslash, /$ and $\wedge$. Whereas for $\wedge$, there is no hope in improving this situation, for residuals there is: for example, if we can ensure that $a = \gamma(a), b = \gamma(b)$ entails $a \cdot b = \gamma(a \cdot b)$, then Lemma 24 will ensure that also residuals are γ-closed, and so canonical operations preserve closure and completeness for **L1** follows by a simple argument. This is exactly what we will do to prove completeness for (finite) relation models. Another way (which we did not work out here) would be to construct canonical models in a way that makes sure that $\gamma(a/b) = \gamma(a)/\gamma(b)$, $\gamma(b \backslash a) = \gamma(b) \backslash \gamma(a)$; in this case, the above argument would easily work and yield completeness.

So the two lemmas of this section show two things: the first one shows that many cases in the crucial lemmas of language/relation completeness follow from general principles, whereas the second one shows the really crucial cases – those involving $/, \backslash$ – and that they cannot be deduced from more general principles.

This is the road map for how to reach the two central results of regular language and finite relation completeness.

7 From ACL to Finite Relation Models

In section 6 of this article, we discussed that under certain conditions, one can get rid of nuclei while preserving truth/falsity of inequations. In our particular case this means: we depart from ACL-semantics and try to get rid of the closure operators, thereby extending completeness to relation models. Note however that in this case, we have to use the relational component of concepts rather than the string-component. As we have completeness for $SCL(REG)$ and consequently for $ACL(FIN)$ by the Isomorphism Theorem, this way of proving completeness for relation models (which has been shown by **(author?)** [2] and **(author?)** [24]) entails completeness for *finite* relation models, which was an open problem up to date.

Canonical relation models interpret $\bullet$ as relation composition, denoted by '**;**'. In this semantics, $\wedge$ and $\vee$ interpreted as $\cap$ and $\cup$, respectively, and residuals are thereby defined implicitly. It is easy to see that there is no hope to get the result for **FL**, as relational semantics is set-theoretical and in **FL** we do not have $\wedge$-$\vee$-distributivity. However, there might be completeness for $\mathbf{L}, \mathbf{L1}$ and $\mathbf{L1}(\vee)$.

The fact that we have to depart from the right-hand side of the Galois connection (namely the relational component) comes however with some additional, fundamental problems: as the relations in ACL are the *right* components of concepts, the order is generally *inverted*. This makes the use of $\vee$ and $\cup$ problematic, as in all residuated structures with $\vee$, we have $m \cdot (m \vee o) = (m \cdot n) \vee (m \cdot o)$, but in general, $m \cdot (n \wedge o) \neq (m \cdot n) \wedge (m \cdot o)$. In ACL, $\vee$ is however exactly interpreted as $\cap$ on the right component, which does not distribute over relation composition (see Lemma 19). But even skipping $\vee$ does solve this problem, as via the residuals $/, \backslash$, the operation $\cdot$ is connected to the underlying order. This problem of the inverted order will actually be the hardest part for our proof of relation model completeness; we will solve it by embedding relation composition into **relation addition** (see below).

Before we address this problem, there is another fundamental issue. Recall that for automatic concepts $(M, R), (N, S)$,

$$(M, R) \circ (N, S) = ((MN)^{\blacktriangleright\blacktriangleleft}, (MN)^{\blacktriangleright}).$$

We have seen that $\circ$ in ACL is intuitively related to relation composition in the right component, however it *cannot* be completely reduced to operations on relations in the general case. This changes if we restrict attention to automata of the form $U(L)$, as in this case Lemma 21 holds, ensuring that

$$(12) \quad (MN)^{\blacktriangleright} = M^{\blacktriangleright};N^{\blacktriangleright}$$

This is the key prove completeness for $\mathbf{L1}, \mathbf{L}$ and (finite) relation models, but not for logics involving $\wedge$ or $\vee$. We have sketched the proof technique in sec-

tion 6: we show that we can substitute $[-]^{\blacktriangleleft\blacktriangleright}$-closed operations with canonical relational operations, and show that for the case of universal automata, the two yield the same result. Recall that for $[-]^{\blacktriangleright}, [-]^{\blacktriangleleft}$ defined with respect to $U(L)$, $(X,Y),(X',Y') \in \mathit{fact}(L)$, $((X,Y),(X',Y')) \in M^{\blacktriangleright}$ iff $XMY' \subseteq L$. Note that maximal left/right factors are always $[-]^{\triangleright\triangleleft}$-closed sets, because for $(X,Y) \in \mathit{fact}(L)$, $X = \{(\epsilon, y) : y \in Y\}^{\triangleleft}$; same for Y. Moreover, recall that Lemma 21 was actually surprisingly strong, as it entails that compositions of arbitrary closed sets are already closed, as long as we restrict ourselves to universal automata.[7] This is what we will do in the sequel; the Isomorphism Theorem makes sure this is sufficient for completeness. We now consider the residuals. Recall that the residuals over relations R, S and $\mathbf{;}$ are defined as

$$R/S = \{(x,y) \in Q \times Q : \{(x,y)\}\mathbf{;}S \subseteq R\} \tag{13}$$

$$S\backslash R = \{(x,y) \in Q \times Q : S\mathbf{;}\{(x,y)\} \subseteq R\} \tag{14}$$

$[-]^{\blacktriangleright}, [-]^{\blacktriangleleft}$ establish an antitone Galois connection, which means that $M/N \circ N \subseteq M$ iff $(M/N \circ N)^{\blacktriangleright} \supseteq M^{\blacktriangleright}$, which (given that we henceforth restrict attention to automata of the form $U(L)$) holds iff $(M/N)^{\blacktriangleright}\mathbf{;}N^{\blacktriangleright} \supseteq M^{\blacktriangleright}$ – the order being inverted! – in virtue of Lemma 21. However, if we just invert the order $\subseteq$ to $\supseteq$ in the definition of residuals as in (13), the resulting set is not guaranteed to exist, and if some such set exists, it is not guaranteed to be unique, as can be easily checked by simple examples.

So the problem arises from the fact that for right components in a Galois connection, the order is inverted, and residuals are defined with respect to the underlying order. Thus if we want to get a relational model based on the right component of ACL-structures, we have to provide a relational operation which allows for a kind of residuation defined with respect to the order $\supseteq$. We do this via **relational addition**, which is defined as follows: fix an underlying carrier set Q, and take $R, R' \subseteq Q \times Q$. Then

$$R \dagger R' := \{(x,y) : \forall z \in Q, \text{ either } (x,z) \in R \text{ or } (z,y) \in R'\} \tag{15}$$

For extensive treatment of relational addition, check [20]. Note that $\dagger$ is defined only with respect to a certain underlying set Q; changing the set, we change the outcome of the operation (similar to complementation $\overline{[-]}$). Still, we often leave Q implicit (as one is used to do for set-theoretic complementation). Relational addition is connected to composition by the following equality, $\overline{[-]}$ and $\dagger$ being defined with respect to the same underlying carrier set Q here and throughout:

$$R \dagger R' = \overline{\overline{R}\mathbf{;}\overline{R'}} \tag{16}$$

This means that $\overline{R\mathbf{;}R'} = \overline{R} \dagger \overline{R'}$. Obviously, the operation $\dagger$ is associative, so for $\mathbf{R} \subseteq \wp(Q \times Q)$, $(Q, \mathbf{R}, \dagger, 1_{\mathbf{R}})$ is a monoid, where Q is explicitly specified in order to make $\dagger$ uniquely defined, and $1_{\mathbf{R}}$ is neutral for relational addition of relations

[7] Recall that sets of the form $M^{\triangleright}$ etc. are always closed.

in $\mathbf{R}$. Here a note is in order: as id – the identity relation – is generally neutral for relation composition, div is generally neutral for relation addition, where

$$(17)\quad \mathsf{div}_Q := \{(x,y) : x,y \in Q, x \neq y\} = \overline{\mathsf{id}_Q}$$

is the diversity relation over Q. However, in the same way as for a set of relations $\mathbf{R}$, there might be a relation $1_{\mathbf{R}}$ which is neutral for composition with relations in $\mathbf{R}$ which however is not an identity relation, there are relations which are neutral for addition with all $R \in \mathbf{R}$, yet are not diversity relations. This hinges on properties of $\mathbf{R}$, and examples can be easily constructed. We therefore have a more general notion of relation monoids and additive relation monoids, where $1_{\mathbf{R}}$ must be neutral, but not the identity or diversity, respectively. We now show that we can isomorphically map a relation monoid $(\mathbf{R}, ;, 1_{\mathbf{R}})$ to a monoid $(Q, \mathbf{R}', \dagger, 1_{\mathbf{R}'})$. Define the map $\psi : \wp(Q \times Q) \to \wp(\wp(Q) \times \wp(Q))$ (i.e. ψ maps relations over Q to relations over $\wp(Q)$) by

$$(18)\quad \psi(R) := \{(M,N) : (M \neq \emptyset \;\&\; \emptyset \neq N) \Longrightarrow \exists x \in M, \exists y \in \overline{N} : (x,y) \in R\}$$

Note that we thus have $(M,\emptyset),(\emptyset,M)$ in *all* relations of the form $\psi(R)$, and hence $\psi(\emptyset) = \{(M,\emptyset),(\emptyset,M) : M \subseteq Q\}$. This is important because $\emptyset$ is not absorbing for $\dagger$: we have, for example, $(Q \times Q) \dagger \emptyset = Q \times Q$, whereas in general, $R;\emptyset = \emptyset$. We now show that ψ is a $\subseteq$-preserving bijection:

Lemma 26 *$R \subseteq R'$ iff $\psi(R) \subseteq \psi(R')$.*

Proof. *Only if*: Assume $R \subseteq R'$, and $(M,N) \in \psi(R)$. Then if $M, N \neq \emptyset$ (otherwise $(M,N) \in \psi(R')$ is obvious), there is $x \in M, y \in \overline{N}$ such that $(x,y) \in R \subseteq R'$, and hence $(M,N) \in \psi(R')$.

If: Contraposition: assume $R \not\subseteq R'$. Then there is $(x,y) \in R, (x,y) \notin R'$. We then have $(\{x\}, Q - \{y\}) \in \psi(R)$, but $(\{x\}, Q - \{y\}) \notin \psi(R')$, as can be easily checked. □

So we have $(x,y) \in R$ iff $(\{x\}, Q - \{y\}) \in \psi(R)$. There is one detail to keep in mind: let id_Q be the identity on a set Q. Then

$$(19)\quad \psi(\mathsf{id}_Q) = \{(M,N) : (M \neq \emptyset \neq N) \Longrightarrow M \not\subseteq N\} \neq \mathsf{div}_{\wp(Q)}$$

We now show that nonetheless ψ isomorphically maps relation composition to relation addition; this is no contradiction, as this only shows that for relations of the form $\psi(R)$, we have $\psi(\mathsf{id}) \dagger \psi(R) = \psi(R) \dagger \psi(\mathsf{id}) = \psi(R)$.

Lemma 27 *For $R, R' \subseteq Q \times Q$, $\psi(R;R') = \psi(R) \dagger \psi(R')$, where $\dagger$ is defined with respect to $\wp(\wp(Q) \times \wp(Q))$.*

Proof. $\subseteq$ Assume $(M,N) \in \psi(R;R')$. Then there is $(x,y) \in R;R'$ such that $x \in M, y \in \overline{N}$. So there is a z such that $(x,z) \in R, (z,y) \in R'$. Hence we have $(M,X) \in R$ for all $X : z \notin X$ and $(Y,N) \in R'$ for all $Y : z \in Y$. Now as the underlying set is $\wp(Q)$, and for all $Z \in \wp(Q)$, we either have $z \in Q$ or $z \notin Q$, we have either $(M,Z) \in \psi(R)$ or $(Z,N) \in \psi(R')$, and hence $(M,N) \in \psi(R) \dagger \psi(R')$.

$\supseteq$ Contraposition: assume $(M,N) \notin \psi(R;R')$. This entails that both $M \neq \emptyset$ and $N \neq \emptyset$; consequently, if $x \in M, y \in \overline{N}$, then $(x,y) \notin R;R'$. So if $x \in M, y \in \overline{N}$, then there is no z such that $(x,z) \in R$ and $(z,y) \in R'$. Put differently: $x \in M, y \in \overline{N}$ and $(x,z) \in R$ entail that $(z,y) \notin R'$.

Now put $O := \{z : (x,z) \in R \ \& \ x \in M\}$. $(M,O) \notin \psi(R)$, because for any $x \in M$ there is no $z \in \overline{O}$ such that $(x,z) \in R$. Conversely, we also have $(O,N) \notin \psi(R')$, because for all $z \in O$, $y \in \overline{N}$, $(z,y) \notin R'$ by assumption and construction of O. Hence $(M,N) \notin \psi(R) \dagger \psi(R')$, and the claim follows. $\Box$

We can now extend Lemma 21 to relation addition: we define new maps $\blacktriangleright', \blacktriangleleft'$, where we put

$$M^{\blacktriangleright'} = \psi(M^{\blacktriangleright}) \tag{20}$$

$$R^{\blacktriangleleft'} = \psi^{-1}(R)^{\blacktriangleleft} \tag{21}$$

$$(M,R) \circ (M,S) = ((MN)^{\blacktriangleright\blacktriangleleft}, R \dagger S) \tag{22}$$

((21) is well-defined because ψ is 1. injective, and here ψ^{-1} only applies to ψ-images). The resulting structure turns out to be isomorphic to $ACL(U(L))$; this is because more than a bijection, ψ is a *pointwise map*, that is,

$$\psi(R) = \bigcup_{(x,y) \in R} \psi\{(x,y)\} \tag{23}$$

This is easy to check: $\supseteq$ follows immediately from Lemma 26, for $\subseteq$, assume that $(M,N) \in \psi(R)$. Then there is $x \in M, y \in \overline{N}$ such that $(x,y) \in R$, hence $(M,N) \in \psi\{(x,y)\}$. (23) ensures that ψ distributes over $\cup, \cap$. What we have gained is the following: we can now define residuals with respect to $\dagger$ and define them with respect to the order $\supseteq$ rather than $\subseteq$. This is based on the following observation:

$$\bigcap_{R \in \mathbf{R}} (R \dagger S) = \overline{\overline{\bigcap_{R \in \mathbf{R}} (R \dagger S)}} = \overline{\bigcup_{R \in \mathbf{R}} (\overline{R};\overline{S})} = \overline{(\bigcup_{R \in \mathbf{R}} \overline{R});\overline{S}} = \bigcap \mathbf{R} \dagger S \tag{24}$$

$$\bigcap_{R \in \mathbf{R}} (S \dagger R) = \overline{\overline{\bigcap_{R \in \mathbf{R}} (S \dagger R)}} = \overline{\bigcup_{R \in \mathbf{R}} (\overline{S};\overline{R})} = \overline{\overline{S};(\bigcup_{R \in \mathbf{R}} \overline{R})} = S \dagger \bigcap \mathbf{R} \tag{25}$$

Both can be also proved set-theoretically, but as we see they also easily follow from the duality of $\dagger$, **;** and $\cap$, $\cup$, respectively. From this it follows that for all relations R, S, there is a *unique smallest* (not largest!) relation $R/_\dagger S$ such that $(R/_\dagger S) \dagger S \supseteq R$; same for $S\backslash_\dagger R$. We call these the **additive residuals** for relations, which *always* (because $(Q \times Q) \dagger \emptyset = Q \times Q = \emptyset \dagger Q \times Q$) and *uniquely* (because of (24),(25)) exist. We can also define them explicitly as follows:

$$R/_\dagger S = \{(x,y) : \text{ for some } z \in Q, \ (x,z) \in R, (y,z) \notin S\}, \tag{26}$$

$$S\backslash_\dagger R = \{(x,y) : \text{ for some } z \in Q, \ (z,y) \in R, (z,x) \notin S\} \tag{27}$$

(Note by the way that, contrary to $\dagger$, $/_\dagger$ and $\backslash_\dagger$ are independent of the underlying set Q.) One can check that this explicit definition coincides with the abstract

characterization as $\supseteq$-residuals for $\dagger$. Consequently, we can define an additive residuated relation monoid as a structure $(Q, \mathbf{R}, \dagger, /_\dagger, \backslash_\dagger, 1_{\mathbf{R}}, \supseteq)$, where for all $R \in \mathbf{R}$, $R \subseteq Q \times Q$, residuals are defined w.r.t. to $\dagger$ and the order $\supseteq$, and $\mathbf{R}$ is closed under $\dagger, /_\dagger, \backslash_\dagger$. We call the class of additive residuated relation monoids $ARRM$. we denote the class of additive residuated relation monoids such that for all $R \in \mathbf{R}$, R is finite *and* $\mathbf{R}$ is finite, by $FINARRM$. The following simple result makes sure the map ψ extends to residuals:

Lemma 28 *Let $(M, \cdot, /, \backslash, 1, \leq)$, $(M', \cdot', /, \backslash, 1', \leq')$ be two residuated monoids, $i : M \to M'$ a bijection such that $i(m \cdot n) =' i(m) \cdot' i(n)$ and $m \leq n$ iff $i(m) \leq' i(n)$. Then $i(m/n) =' i(m)/i(n)$, $i(m\backslash n) =' i(m)\backslash i(n)$.*[8]

Proof. $\leq' i(x)/i(y)$ is the largest element such that $i(x)/i(y) \cdot' i(y) \leq' i(x)$. We have $x/y \cdot y \leq x$ iff $i(x/y \cdot y) \leq' i(x)$ iff $i(x/y) \cdot' i(y) \leq' i(x)$. Hence $i(x/y) \leq' i(x)/i(y)$.

$\geq'$ Firstly note that all mentioned properties of i apply equally to i^{-1} (easy exercise to show). Now assume $z \leq' i(x)/i(y)$. We have $z \leq' i(x)/i(y)$ iff $z \cdot' i(y) \leq' i(x)$ iff $i^{-1}(z \cdot' i(y)) \leq x$ iff $i^{-1}(z) \cdot y \leq x$ iff $i^{-1}(z) \leq x/y$. Hence $i(i^{-1}(z)) =' z \leq' i(x/y)$; and so it follows that $i(x)/i(y) \leq' i(x/y)$. □

Lemma 29 *Assume that $[-]^{\blacktriangleright}, [-]^{\blacktriangleleft}$ are defined w.r.t. some automaton $U(L)$, and $[-]^{\blacktriangleright'} = \psi \circ [-]^{\blacktriangleright}$, $[-]^{\blacktriangleleft'} = [-]^{\blacktriangleleft} \circ \psi^{-1}$. Then $R^{\blacktriangleleft'\blacktriangleright'} \dagger S^{\blacktriangleleft'\blacktriangleright'} \supseteq (R \dagger S)^{\blacktriangleleft'\blacktriangleright'}$, that is, $[-]^{\blacktriangleleft'\blacktriangleright'}$ is a nucleus for the operation $\dagger$ and the order $\supseteq$.*

Proof. We obviously have $R^{\blacktriangleleft'\blacktriangleright'} \dagger S^{\blacktriangleleft'\blacktriangleright'} \supseteq R \dagger S$, because $\dagger$ is monotonous in its arguments. We show that $R^{\blacktriangleleft'\blacktriangleright'} \dagger S^{\blacktriangleleft'\blacktriangleright'}$ is a closed set: we have

$$\begin{aligned} R^{\blacktriangleleft'\blacktriangleright'} \dagger S^{\blacktriangleleft'\blacktriangleright'} &= \psi(\psi^{-1}(R)^{\blacktriangleleft\blacktriangleright}) \dagger \psi(\psi^{-1}(S)^{\blacktriangleleft\blacktriangleright}) \\ &= \psi(\psi^{-1}(R)^{\blacktriangleleft\blacktriangleright}; \psi^{-1}(S)^{\blacktriangleleft\blacktriangleright}), \end{aligned}$$

where $\psi^{-1}(R)^{\blacktriangleleft\blacktriangleright}; \psi^{-1}(S)^{\blacktriangleleft\blacktriangleright})$ is closed w.r.t. $[-]^{\blacktriangleleft\blacktriangleright}$ by Lemma 21. Moreover, the ψ-image of a $[-]^{\blacktriangleleft\blacktriangleright}$-closed set by definition is a $[-]^{\blacktriangleleft'\blacktriangleright'}$-closed set. As $(R \dagger S)^{\blacktriangleleft'\blacktriangleright'}$ is the smallest closed set containing $(R \dagger S)$, it follows that $R^{\blacktriangleleft'\blacktriangleright'} \dagger S^{\blacktriangleleft'\blacktriangleright'} \supseteq (R \dagger S)^{\blacktriangleleft'\blacktriangleright'}$. □

These last lemmas entail the following:

Corollary 30 *Let $[-]^{\blacktriangleright'}, [-]^{\blacktriangleleft'}$ be defined as in Lemma 29, and assume in addition that $(MN)^{\blacktriangleright'} = M^{\blacktriangleright'} \dagger N^{\blacktriangleright'}$, and $M = M^{\blacktriangleright'\blacktriangleleft'}, N = N^{\blacktriangleright'\blacktriangleleft'}$. Then $(M/N)^{\blacktriangleright'} = M^{\blacktriangleright'}/_\dagger N^{\blacktriangleright'}$, $(M\backslash N)^{\blacktriangleright'} = M^{\blacktriangleright'}\backslash_\dagger N^{\blacktriangleright'}$.*

Proof. The map $[-]^{\blacktriangleright'}$ is obviously a bijection from $[-]^{\blacktriangleright'\blacktriangleleft'}$-closed sets to $[-]^{\blacktriangleleft'\blacktriangleright'}$-closed sets. Hence it satisfies all premises of Lemma 28, and by Lemmas 24,29, $\dagger$-residuals of $[-]^{\blacktriangleleft'\blacktriangleright'}$ closed sets are already $[-]^{\blacktriangleleft'\blacktriangleright'}$-closed. □

Actually, with the results we have it is not difficult to establish the equalities $(M/N)^{\blacktriangleright'} = M^{\blacktriangleright'}/_\dagger N^{\blacktriangleright'}$ etc. by purely set-theoretic means either. Note that this result entails that for closed sets of strings M, N, we have $M/N =$

[8] Keep in mind that here $m =' n$ is an abbreviation for $m \leq' n$ & $n \leq' m$, same for $m = n$.

$(M^{\blacktriangleright'}/N^{\blacktriangleright'})^{\blacktriangleleft'}$, $M\backslash N = (M^{\blacktriangleright'}\backslash N^{\blacktriangleright'})^{\blacktriangleleft'}$; and dually for closed R, S, $R^{\blacktriangleleft'}/S^{\blacktriangleleft'} = (R/_\dagger S)^{\blacktriangleleft'}$ and $R^{\blacktriangleleft'}\backslash S^{\blacktriangleleft'} = (R\backslash_\dagger S)^{\blacktriangleleft'}$.

Now we easily get a completeness result of **L1** for finite relation models, but surprisingly, these models have the form $(Q, \mathbf{R}, \dagger, /_\dagger, \backslash_\dagger, 1_{\mathbf{R}} \supseteq)$; importantly, the underlying partial order which defines the residuals is *not* $\subseteq$, but rather $\supseteq$. To make things a bit more explicit, we define the notion of an **additive relational concept monoid** as a structure $ARCM(\phi) = (\mathcal{B}_\phi, \circ, /, \backslash, 1)$, where we have $[-]^{\blacktriangleright} : \wp(\Sigma^*) \to \wp(Q \times Q)$,$[-]^{\blacktriangleleft} : \wp(Q \times Q) \to \wp(\Sigma^*)$ defined as in ACL by $M^{\blacktriangleright} = \bigcap_{w \in M} \phi(w)$, $R^{\blacktriangleleft} = \{w : \phi(w) \supseteq R\}$. Then $\mathcal{B}_\phi$ is the set of pairs (M, R) where $M^{\blacktriangleright} = R$, $R^{\blacktriangleleft} = M$; $1 = (\{\epsilon\}^{\blacktriangleright\blacktriangleleft}, \{\epsilon\}^{\blacktriangleright})$. But now things differ:

$$(M, S) \circ (N, S) = ((R \dagger S)^{\blacktriangleleft}, R \dagger S),$$
$$(M, R)/(N, S) = ((R/_\dagger S)^{\blacktriangleleft}, R/_\dagger S),$$
$$(M, S)\backslash(N, S) = ((R\backslash_\dagger S)^{\blacktriangleleft}, R\backslash_\dagger S).$$

Hence we now have the "main action" in the right component rather than in the left! The reason this works is that we have used $\dagger$, for which we can define residuals for the inverted order $\supseteq$ rather than $\subseteq$, so this was really the crucial step. The next lemma is now straightforward:

Lemma 31 *For every language L, we can construct a $\chi : \Sigma^* \to \wp(Q \times Q)$ such that $ACL_{\boldsymbol{L1}}(U(L)) \cong ARCM(\chi)$.*

Proof. Assume $U(L) = \langle \phi, \Sigma, F \rangle$. Then we put $\chi = \psi \circ \phi$, and the claim easily follows from Lemmas 21, 27 and 30. □

Note that this lemma cannot be easily extended to $\vee$: the reason is that the union of closed sets is no longer necessarily closed, so we lose some of the premises of Lemma 30 which are necessary for its proof.

Corollary 32 *For every language L, we can construct a $\boldsymbol{B} \in ARRM$ such that $ACL_{\boldsymbol{L1}}(U(L)) \cong \boldsymbol{B}$.*

Proof. Straightforward consequence of the previous lemma, as in $ARCM$, all operations can be reduced to operations of $ARRM$. □

Theorem 10 *We have $\Vdash_{\boldsymbol{L1}} \Gamma \vdash \alpha$ iff $ARRM \models \Gamma \vdash \alpha$ iff $FINARRM \models \Gamma \vdash \alpha$.*

We could also state a corresponding result for **L**, but we skip in order to not drown in notation (we return to **L** in Theorem 12).

Proof. Soundness is clear. For completeness, we use completeness for ACL and regular languages: if $L \in REG$, then $ACL(U(L))$ is finite, and the set of right components of concepts form a finite set of finite relations.

So assume we have $\nVdash_{\mathbf{L1}} \Gamma \vdash \alpha$. Then there is a finite model $ACL_{\mathbf{L1}}(U(L))$, interpretation σ such that $ACL_{\mathbf{L1}}(U(L)), \sigma \not\models \Gamma \vdash \alpha$. By the previous lemma and corollary, we have a $\mathbf{B} \in ARMM$ such that $\mathbf{B} \cong ACL_{\mathbf{L1}}(U(L))$. Let $\eta : \mathbf{B} \to ACL_{\mathbf{L1}}(U(L))$ denote the corresponding isomorphism. We thus have $\mathbf{B}, \eta \circ \sigma \not\models \Gamma \vdash \alpha$. □

Residuated relation monoids and additive residuated relation monoids are obviously closely related by the duality of **;** and $\dagger$. This duality can be extended to residuals:

Lemma 33 *For all sets Q, $R, S \subseteq Q \times Q$, $R/S = \overline{\overline{R}/_\dagger \overline{S}}$ and $R\backslash S = \overline{\overline{R}\backslash_\dagger \overline{S}}$.*

Proof. To see this, just consider that the law of residuation applies to the two of them, with respective monoid-operations and orders. So we have $X \subseteq R/S$ iff $X;S \subseteq R$ iff $\overline{X;S} \supseteq \overline{R}$ iff $\overline{X} \dagger \overline{S} \supseteq \overline{R}$ iff $\overline{X} \supseteq \overline{R}/_\dagger \overline{S}$ iff $X \subseteq \overline{\overline{R}/_\dagger \overline{S}}$. Same for $\backslash_\dagger$. □

Now the following is quite simple:

Lemma 34 *The map $\overline{[-]}$ (complement with respect to Q) is an isomorphism between $(Q, \boldsymbol{R}, \dagger, /_\dagger, \backslash_\dagger, 1_{\boldsymbol{R}}, \supseteq)$ and $(\{\overline{R} : R \in \boldsymbol{R}\}, ;, /, \backslash, \overline{1_{\boldsymbol{R}}}, \subseteq)$.*

Proof. We have already seen that

- $\overline{R \dagger S} = \overline{R};\overline{S}$
- $\overline{R/_\dagger S} = \overline{R}/\overline{S}$
- $\overline{R\backslash_\dagger S} = \overline{R}\backslash\overline{S}$

This entails that $\overline{1_{\mathbf{R}}}$ is neutral for $\{\overline{R} : R \in \mathbf{R}\}$ and $;$. Finally, it is obvious that 1. $\overline{[-]}$ is a bijection, and 2. $\overline{R} \supseteq \overline{S}$ iff $R \subseteq S$. □

Given $\mathbf{B} \in ARRM$, we denote its $\overline{[-]}$-image by $\overline{\mathbf{B}}$. Now it is straightforward to prove completeness for **L1** and finite canonical relation models, which is the most important result of this section:

Theorem 11 *We have $\Vdash_{\boldsymbol{L1}} \Gamma \vdash \alpha$ iff $\mathsf{Rel}(RM) \models \Gamma \vdash \alpha$ iff $\mathsf{FinRel}(RM) \models \Gamma \vdash \alpha$.*

Proof. Soundness is again clear. Assume $\nVdash_{\mathbf{L1}} \Gamma \vdash \alpha$. Then there is a (finite) additive relation monoid $\mathbf{B} \in ARRM$, interpretation $\overline{\sigma}$ such that $\mathbf{B}, \sigma \not\models \Gamma \vdash \alpha$. $\mathbf{B}$ is isomorphic to its dual model $\overline{\mathbf{B}}$, where $\overline{\mathbf{B}} \in \mathsf{Rel}(RM)$ and $\overline{\overline{\sigma}(\Gamma)} \subseteq \overline{\overline{\sigma}(\alpha)}$ iff $\overline{\sigma}(\Gamma) \supseteq \overline{\sigma}(\alpha)$, in virtue of the previous lemma. Hence $\overline{\mathbf{B}}, \overline{[-]} \circ \sigma \not\models \Gamma \vdash \alpha$. By completeness for $FINARRM$, there is also a finite model $\mathbf{B} \in FINARRM$, and hence $\overline{\mathbf{B}} \in \mathsf{FinRel}(RM)$ (the underlying carrier set being finite), which is such that $\overline{\mathbf{B}}, \overline{[-]} \circ \sigma \not\models \Gamma \vdash \alpha$. □

We have to adapt the proof to **L** and $\mathsf{FinRel}(RS)$. The main difference is that in this case, we have to specify a maximal relation R_{max}. As the proof departs from some nuclear (semi)-automaton $\mathcal{A}$ and $ACL(\mathcal{A})$, for our purposes we do the following: given a semi-automaton $\mathcal{A} = \langle \Sigma, \phi \rangle$, we put

$$R^{\mathcal{A}}_{max} = \bigcup_{w \in \Sigma^+} \phi(w)$$

This means that we consider all transitions in the automaton, where we might also have transitions of the form (x, x), but we do not generally have the full identity relation on the set of states id_Q, as $\mathsf{id}_Q = \phi(\epsilon)$. Now it is easy to see that all intermediate steps remain equally valid in this case (though the unit is no longer present): Lemma 21 works as before, this time considering closed sets in $ACL^+(\mathcal{A})$). The lemmas regarding the bijection of $(\mathbf{R}, ;)$ and $(\psi[\mathbf{R}], \dagger)$ work

equally well,[9] we just have to take care that with every mapping, R_{max} is also mapped accordingly. Lemmas 28 and 30 work equally well in absence of a unit element, and also the remaining intermediate steps can be taken over without any problem, simply skipping the (implicit) requirement that there be a unit, as it is not needed at any place. So we have the following:

Theorem 12 $\Vdash_{\boldsymbol{L}} \Gamma \vdash \alpha$ *iff* $\mathsf{Rel}(RS) \models \Gamma \vdash \alpha$ *iff* $\mathsf{FinRel}(RS) \models \Gamma \vdash \alpha$.

So we have settled the problem of completeness for finite relation models for $\mathbf{L}, \mathbf{L1}$, left open by **(author?)** [24] and **(author?)** [2].

8 Conclusion

We have presented a variety of completeness results for the Full Lambek calculus and its various fragments. These comprised established results such as completeness results for syntactic concept lattices, but also new results such as completeness for finite relation models. What is common for all models we have considered is that they are "language-theoretic" in a broad sense. Whereas one might claim this is not true for relational semantics, the automata-theoretic semantics clearly shows that it is, and indeed we have proved completeness for finite relation models only via formal language theory.

All our results are based on the construction of closure operators, Galois connections and nuclei: these allow us to give rather simple proofs for completeness, whereas other constructions such as embeddings and quasi-embeddings easily allow us to extend results to interpretations which do not involve closure operators. This shows the usefulness of Galois connections in the context of substructural logics and formal language theory.

One open problem we want to mention is the following: despite the fact that $\vee$ in connection with $\bullet, /, \backslash$ is rather well-behaved, we did not find it possible to extend completeness for relation models from $\mathbf{L1}$ to $\mathbf{L1}(\vee)$ (where $\vee$ is interpreted as set-theoretic $\cup$), despite the fact that often this result looks "at hand". So can the methods used here be extended to yield completeness results for relational semantics and $\mathbf{L1}(\vee)$?

Another prominent open problem is the following: in a similar way as we have proved completeness for relation models from automaton models, it might be possible to prove completeness for canonical language models (L-models). Though this result is already established, in our case it could immediately be strengthened to regular languages, which is still an open problem to our knowledge.

[9] This is because ψ is a pointwise map, see (23).

References

[1] Hajnal Andréka and Szabolcs Mikulás. Lambek calculus and its relational semantics: Completeness and incompleteness. *Journal of Logic, Language, and Information*, 3:1–37, 1994.

[2] Carolyn Brown and Doug Gurr. Relations and non-commutative linear logic. *Journal of Pure and Applied Algebra*, 105(2):117 – 136, 1995.

[3] Wojciech Buszkowski. Compatibility of a categorial grammar with an associated category system. *Mathematical Logic Quarterly*, 28(14-18):229–238, 1982.

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

[5] Wojciech Buszkowski. Algebraic structures in categorial grammar. *Theor. Comput. Sci.*, 1998(1-2):5–24, 1998.

[6] 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.

[7] 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.

[8] 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.

[9] Alexander Clark. The syntactic concept lattice: Another algebraic theory of the context-free languages? *J. Log. Comput.*, 25(5):1203–1229, 2015.

[10] B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, Cambridge, 2 edition, 1991.

[11] Maciej Farulewski. On finite models of the Lambek calculus. *Studia Logica*, 80(1):63–74, 2005.

[12] Nikolaos Galatos, Peter Jipsen, Tomasz Kowalski, and Hiroakira Ono. *Residuated Lattices: An Algebraic Glimpse at Substructural Logics*. Elsevier, 2007.

[13] Zellig S. Harris. *Structural Linguistics*. The University of Chicago Press, 1963.

[14] Makoto Kanazawa. The Lambek calculus enriched with additional connectives. *Journal of Logic, Language, and Information*, 1:141–171, 1992.

[15] Stepan Kuznetsov. Conjunctive grammars in greibach normal form and the lambek calculus with additive connectives. In Glyn Morrill and Mark-Jan Nederhof, editors, *Formal Grammar - 17th and 18th International Conferences, FG 2012, Opole, Poland, August 2012, Revised Selected Papers, FG 2013, Düsseldorf, Germany, August 2013. Proceedings*, volume 8036 of *Lecture Notes in Computer Science*, pages 242–249. Springer, 2013.

[16] Joachim Lambek. The mathematics of sentence structure. *The American Mathematical Monthly*, 65:154–169, 1958.

[17] Joachim Lambek. On the calculus of syntactic types. In Roman Jakobson, editor, *Structure of Language and its Mathematical Aspects*, pages 166–178. Providence, 1961.

[18] Hans Leiß. Learning context free grammars with the finite context property: A correction of a. clark's algorithm. In Glyn Morrill, Reinhard Muskens, Rainer Osswald, and Frank Richter, editors, *Formal Grammar - 19th International Conference, FG 2014, Tübingen, Germany, August 16-17, 2014. Proceedings*, volume 8612 of *Lecture Notes in Computer Science*, pages 121–137. Springer, 2014.

[19] Sylvain Lombardy and Jacques Sakarovitch. The universal automaton. In Jörg Flum, Erich Grädel, and Thomas Wilke, editors, *Logic and Automata: History and Perspectives [in Honor of Wolfgang Thomas].*, volume 2 of *Texts in Logic and Games*, pages 457–504. Amsterdam University Press, 2008.

[20] Roger Maddux. *Relation Algebras.* Elsevier, Amsterdam et.al., 2006.

[21] Glyn Morrill, Oriol Valentín, and Mario Fadda. The displacement calculus. *Journal of Logic, Language and Information*, 20(1):1–48, 2011.

[22] Mitsuhiro Okada and Kazushige Terui. The finite model property for various fragments of intuitionistic linear logic. *Journal of Symbolic Logic*, 64(2):790–802, 1999.

[23] 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.

[24] Mati Pentus. Models for the Lambek calculus. *Annals of Pure and Applied Logic*, 75:179–213, 1995.

[25] Greg Restall. *An Introduction to Substructural Logics.* Routledge, New York, 2008.

[26] 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.

[27] Johan van Benthem. *Language in Action : Categories, Lambdas and Dynamic Logic*, volume 130 of *Studies in logic and the foundations of mathematics ; 130*. North-Holland, Amsterdam [u.a.], 1991.

[28] 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.

[29] Christian Wurm. Automatic concepts and automata-theoretic semantics for the full lambek calculus. In Maxime Amblard, Philippe de Groote, Sylvain Pogodalla, and Christian Retoré, editors, *Logical Aspects of Computational Linguistics. 2016, Nancy, France, December 5-7, 2016, Proceedings*, volume 10054 of *Lecture Notes in Computer Science*, pages 308–323, 2016.

[30] Christian Wurm. On some extensions of syntactic concept lattices: Completeness and finiteness results. In *Proceedings of Formal Grammar 2015 and 2016*, pages 164–179, 2016.

[31] David N. Yetter. Quantales and (noncommutative) linear logic. *J. Symb. Log.*, 55(1):41–64, 1990.