FULL LAMBEK CALCULUS WITH CONTRACTION IS UNDECIDABLE

KAREL CHVALOVSKÝ AND ROSTISLAV HORČÍK

Abstract. We prove that the set of formulae provable in the full Lambek calculus with the structural rule of contraction is undecidable. In fact, we show that the positive fragment of this logic is undecidable.

1. Introduction

Besides the cut rule, Gentzen's sequent calculus **LJ** for propositional intuitionistic logic contains other structural rules, namely the rules of contraction (c), exchange (e), left weakening (i), and right weakening (o). By removing all these rules from **LJ**, one arrives at the full Lambek calculus **FL**. More generally, every extension of **FL** by a subset of the rules (c), (e), (i), and (o) defines a logic between **FL** and **LJ**. In [8] these logics are called *basic substructural logics*. It is known that each of these logics has an analytic sequent calculus. In particular, the cut rule is eliminable in all these calculi if the contraction rule is introduced in its global variant (see [22] or [8, Chapter 4]).

Cut elimination is closely related to decidability. It is known that all basic substructural logics are decidable except of $\mathbf{FL_c}$ and $\mathbf{FL_{co}}$, where the former is the extension of **FL** by the contraction rule and the latter is the extension of $\mathbf{FL_c}$ by the right weakening rule. The decidability of basic substructural logics without the contraction rule follows immediately from the cut elimination theorem and is proved in [17]. On the other hand, such an easy argument is not applicable for logics with the contraction rule since this rule makes the proof-search tree infinite. Nevertheless, intuitionistic logic is decidable [9, 10] and the same holds for the extension of **FL** by the exchange and contraction rules [16] (the original combinatorial idea from the proof goes back to Kripke [18]). In contrast, we show that $\mathbf{FL_c}$ and $\mathbf{FL_{co}}$ are the only undecidable logics among all basic substructural logics. Actually, we prove that their common positive fragment $\mathbf{FL_c^+}$ is undecidable.

Among known propositional substructural logics, there are not so many logics with an undecidable set of provable formulae. One of them is the relevance logic **R**, which is a fragment of the involutive distributive **FL** with the exchange and contraction rules. The undecidability of its positive fragment is established in [23]. Another example is the extension of **FL** by the modular law. Since the equational theory of modular lattices is undecidable [6], one can easily extend this result to the extension of **FL** by the modular law, as pointed out in [13]. One should also mention the undecidability of propositional linear logic [20]. Nevertheless, its undecidability is caused by the expressive power of exponentials, while the fragment of linear logic without exponentials is PSPACE-complete [20].

The following paragraphs outline our undecidability proof. We start with an undecidable problem P from [12] (see Theorem 3.1), where it is shown that the deducibility problem for $\mathbf{FL_c^+}$ (i.e. the question whether a formula is provable from a finite theory) is undecidable. The problem P is formulated as a reachability question for a string rewriting system simulating a Minsky machine using only square-free words. This ensures that the contraction rule does not affect the simulation of computation.

In order to isolate key ideas of the proof, we refrain from presenting a direct reduction from the problem P into $\mathbf{FL}_{\mathbf{c}}^{+}$. We instead introduce an auxiliary problem and split the reduction into two steps.

First, Section 3.2 shows how to reduce the reachability problem for a string rewriting system to the same problem for an atomic conditional string rewriting system, i.e. a string rewriting system where the rules can have only atoms on the right-hand side and their usage is restricted to specific contexts.

Second, an encoding of the reachability problem for an atomic conditional string rewriting system into $\mathbf{FL}_{\mathbf{c}}^{+}$ is presented in Section 4. A set of rewriting rules is encoded as a lattice conjunction (meet) of implications. The conditionality of rewriting rules is handled by the lattice disjunction (join). The idea of using the lattice disjunction for similar purposes comes from [14], where it is used for linear logic. In fact, our application of this idea was inspired by [3].

The completeness of encoding from Section 4 is proved by a semantical method similar to the one used in [19]. This method relies on a sound and complete algebraic semantics for $\mathbf{FL}_{\mathbf{c}}^{+}$ based on a variety of residuated lattices $\mathcal{RL}_c$. In order not to mix different formalisms, we opt for using an algebraic formalism throughout the paper. Actually, we prove undecidability of the equational theory for $\mathcal{RL}_c$ which immediately implies that $\mathbf{FL}_{\mathbf{c}}^{+}$ is undecidable.

Section 5 contains several comments on possible modifications of the main result, as well as its connection to the deduction theorem.

2. Preliminaries

As was mentioned in the introduction, we show that even the positive fragment of full Lambek calculus with the contraction rule $\mathbf{FL}_{\mathbf{c}}^{+}$ is undecidable. Probably the most natural way of presenting $\mathbf{FL}_{\mathbf{c}}^{+}$ is in terms of a sequent calculus. Formulae are formed in a standard way from a countable set of variables *Var* and a constant 1 using the following connectives: fusion ($\cdot$), two implications ($\backslash$ and $/$), join ($\vee$), and meet ($\wedge$). It should be noted that we have two implications, because there are two natural ways how to obtain them in systems where the rule of exchange is not valid. The set of all formulae is denoted by *Fm*. When writing formulae, we omit some parentheses using the convention that fusion binds tighter than implications followed by meet and join. Furthermore, we use the fact that fusion is associative in $\mathbf{FL}_{\mathbf{c}}^{+}$. Moreover, we often omit fusion completely, i.e. a formula $\varphi \cdot \psi$ is shortly written as $\varphi\psi$.

A sequent is a pair $\Gamma \Rightarrow \varphi$, where Γ is a (possibly empty) sequence of formulae and φ is a formula. The elements of Γ are separated by commas as usual and the intended meaning of these commas is fusion.

Definition 2.1. The sequent calculus for $\mathbf{FL}_{\mathbf{c}}^{+}$ has the following axioms and inference rules[1]:

$$\text{(Id)}\ \frac{}{\varphi \Rightarrow \varphi} \qquad\qquad \text{(Cut)}\ \frac{\Gamma_1, \varphi, \Gamma_2 \Rightarrow \psi \qquad \Delta \Rightarrow \varphi}{\Gamma_1, \Delta, \Gamma_2 \Rightarrow \psi}$$

[1] It is worth noting that we formulate the rule of contraction (c) for sequences and not only for formulae. The rule of contraction only for formulae does not admit cut elimination. However, as we have fusion in the language, both rules are equivalent in the presence of (Cut).

$$(1L)\ \frac{\Gamma_1, \Gamma_2 \Rightarrow \psi}{\Gamma_1, 1, \Gamma_2 \Rightarrow \psi} \qquad (1R)\ \frac{}{\Rightarrow 1}$$

$$(\cdot L)\ \frac{\Gamma_1, \varphi, \psi, \Gamma_2 \Rightarrow \chi}{\Gamma_1, \varphi \cdot \psi, \Gamma_2 \Rightarrow \chi} \qquad (\cdot R)\ \frac{\Gamma \Rightarrow \varphi \qquad \Delta \Rightarrow \psi}{\Gamma, \Delta \Rightarrow \varphi \cdot \psi}$$

$$(\backslash L)\ \frac{\Gamma_1, \varphi, \Gamma_2 \Rightarrow \psi \qquad \Delta \Rightarrow \chi}{\Gamma_1, \Delta, \chi \backslash \varphi, \Gamma_2 \Rightarrow \psi} \qquad (\backslash R)\ \frac{\varphi, \Gamma \Rightarrow \psi}{\Gamma \Rightarrow \varphi \backslash \psi}$$

$$(/L)\ \frac{\Gamma_1, \varphi, \Gamma_2 \Rightarrow \psi \qquad \Delta \Rightarrow \chi}{\Gamma_1, \varphi / \chi, \Delta, \Gamma_2 \Rightarrow \psi} \qquad (/R)\ \frac{\Gamma, \varphi \Rightarrow \psi}{\Gamma \Rightarrow \psi / \varphi}$$

$$(\vee L)\ \frac{\Gamma, \varphi, \Delta \Rightarrow \chi \qquad \Gamma, \psi, \Delta \Rightarrow \chi}{\Gamma, \varphi \vee \psi, \Delta \Rightarrow \chi} \qquad (\vee R)\ \frac{\Gamma \Rightarrow \varphi_i}{\Gamma \Rightarrow \varphi_1 \vee \varphi_2} \text{ for } i = 1, 2$$

$$(\wedge L)\ \frac{\Gamma, \varphi_i, \Delta \Rightarrow \psi}{\Gamma, \varphi_1 \wedge \varphi_2, \Delta \Rightarrow \psi} \text{ for } i = 1, 2 \qquad (\wedge R)\ \frac{\Gamma \Rightarrow \varphi \qquad \Gamma \Rightarrow \psi}{\Gamma \Rightarrow \varphi \wedge \psi}$$

$$(c)\ \frac{\Gamma_1, \Delta, \Delta, \Gamma_2 \Rightarrow \psi}{\Gamma_1, \Delta, \Gamma_2 \Rightarrow \psi}$$

The provability in the sequent calculus for $\mathbf{FL_c^+}$ is defined in the usual way—a proof is a tree labeled by sequents with only axioms in leaves and all the other vertices are obtained from their children by the inference rules. We say that a formula φ is a theorem of $\mathbf{FL_c^+}$ if $\Rightarrow \varphi$ is provable in $\mathbf{FL_c^+}$.

The logic $\mathbf{FL_c^+}$ has a sound and complete algebraic semantics based on residuated lattices. A *residuated lattice* $\mathbf{A} = \langle A, \wedge, \vee, \cdot, \backslash, /, 1 \rangle$ is an algebraic structure such that $\langle A, \wedge, \vee \rangle$ is a lattice, $\langle A, \cdot, 1 \rangle$ is a monoid, and for all $a, b, c \in A$ we have

$$a \cdot b \leq c \quad \text{iff} \quad b \leq a\backslash c \quad \text{iff} \quad a \leq c/b, \tag{1}$$

where $\leq$ is the order induced by the lattice structure of $\mathbf{A}$, i.e. $a \leq b$ iff $a \vee b = b$.

Given a residuated lattice $\mathbf{A}$, an $\mathbf{A}$-*evaluation* (or shortly evaluation if $\mathbf{A}$ is clear from the context) e is a map from Fm into A preserving all the connectives, i.e. it is a homomorphism from the formula algebra on Fm to $\mathbf{A}$. An identity is a pair of formulae $\langle \varphi, \psi \rangle$ written suggestively as $\varphi = \psi$. We say that an identity $\varphi = \psi$ holds in $\mathbf{A}$ if for every $\mathbf{A}$-evaluation e we have $e(\varphi) = e(\psi)$. More generally, an identity $\varphi = \psi$ holds in a class of residuated lattices $\mathcal{K}$ if it holds in every residuated lattice from $\mathcal{K}$. An identity of the form $\varphi \vee \psi = \psi$ is shortly denoted by $\varphi \leq \psi$.

Fact 2.1. *The following identities hold in the class of all residuated lattices:*

- $x(x\backslash y) \leq y,$
- $x(y \vee z) = xy \vee xz,$
- $(y \vee z)x = yx \vee zx.$

Note that $x \leq y$ implies $xz \leq yz$ and $zx \leq zy$ by the distributivity of fusion over join.

A residuated lattice $\mathbf{A}$ is called *square increasing* if the identity $x \leq x^2$ holds in $\mathbf{A}$. It is well known (see e.g. [8]) that the class of square-increasing residuated lattices forms a variety denoted by $\mathcal{RL}_c$, i.e. it is axiomatized by a set of identities.

Theorem 2.2 (e.g. [8]). $\mathbf{FL_c^+}$ *is sound and complete with respect to the class of square-increasing residuated lattices. More precisely, the sequent* $\psi \Rightarrow \varphi$ *is provable in* $\mathbf{FL_c^+}$ *iff the identity* $\psi \leq \varphi$ *holds in* $\mathcal{RL}_c$.

Since the sequents $\Rightarrow \varphi$ and $1 \Rightarrow \varphi$ are equivalent in terms of provability in $\mathbf{FL_c^+}$, it follows that the sequent $\Rightarrow \varphi$ is provable in $\mathbf{FL_c^+}$ iff the identity $1 \leq \varphi$ holds in $\mathcal{RL}_c$. Furthermore, observe that by (1) an identity $\varphi \leq \psi$ holds in $\mathcal{RL}_c$ iff $1 \leq \varphi\backslash\psi$ holds there. Consequently, if we prove that the set of identities of the form $\varphi \leq \psi$ valid in $\mathcal{RL}_c$ is undecidable, then we obtain the same for the set of provable formulae in $\mathbf{FL_c^+}$.

We opt for using algebraic semantics in our proofs because algebraic notation in this case seems to be more compact. Nevertheless, this choice is not essential and has no influence on the construction itself. Moreover, a reader preferring e.g. proof-theoretical notions can adapt even all the proofs, because the main ideas in them remain the very same.

2.1. **Residuated frames.** In the following paragraphs, we recall residuated frames which will be useful in the construction of a suitable countermodel in the proof of completeness of our encoding. We start with an important example of a residuated lattice called the powerset monoid.

Example 2.1 (see e.g. [8]). Let $\mathbf{M} = \langle M, \cdot, 1\rangle$ be a monoid. The *powerset monoid* is the residuated lattice $\mathcal{P}(\mathbf{M}) = \langle \mathcal{P}(M), \cap, \cup, \cdot, \backslash, /, \{1\}\rangle$ defined on the powerset of M, where for $X, Y, Z \subseteq M$ the operations are defined as follows:

$$\begin{aligned} X \cdot Y &= \{\, x \cdot y \in M \mid x \in X,\ y \in X \,\}, \\ X\backslash Z &= \{\, y \in M \mid X \cdot \{y\} \subseteq Z \,\}, \\ Z/Y &= \{\, x \in M \mid \{x\} \cdot Y \subseteq Z \,\}. \end{aligned}$$

Note that $1 \in X\backslash Z$ iff $X \subseteq Z$.

Other examples of residuated lattices can be obtained from the powerset monoid $\mathcal{P}(\mathbf{M})$ by considering a suitable closure operator on the poset $\langle \mathcal{P}(M), \subseteq\rangle$. Recall that a closure operator on $\langle \mathcal{P}(M), \subseteq\rangle$ is a map $\gamma\colon \mathcal{P}(M) \to \mathcal{P}(M)$ such that for all $X, Y \subseteq M$ we have

- $X \subseteq \gamma(X)$,
- $\gamma(\gamma(X)) = \gamma(X)$, and
- $X \subseteq Y$ implies $\gamma(X) \subseteq \gamma(Y)$.

A subset $X \subseteq M$ is said to be γ-*closed* if $X = \gamma(X)$. The set of all γ-closed subsets of M is denoted $\mathcal{P}(M)_\gamma$. Recall that $\langle \mathcal{P}(M)_\gamma, \cap, \cup_\gamma\rangle$ forms a complete lattice where $X \cup_\gamma Y = \gamma(X \cup Y)$.

A subset $\mathcal{B} \subseteq \mathcal{P}(M)_\gamma$ of γ-closed sets is said to be a *basis* for γ if every $X \in \mathcal{P}(M)_\gamma$ can be expressed as the intersection of all the basis elements above X, i.e. $X = \bigcap\{\, B \in \mathcal{B} \mid X \subseteq B \,\}$. Given a basis $\mathcal{B}$ for the closure operator γ, the equivalence

$$X \subseteq \gamma(Y) \quad \text{iff} \quad Y \subseteq B \text{ implies } X \subseteq B \text{ for all } B \in \mathcal{B} \tag{2}$$

holds for all $X, Y \subseteq M$.

It is well known that every closure operator on $\langle \mathcal{P}(M), \subseteq\rangle$ is induced by a binary relation $N \subseteq M \times T$ for some set T (see e.g. [5, 8]). Given such a relation $N \subseteq M \times T$, one can introduce the following two maps which define a Galois connection between $\langle \mathcal{P}(M), \subseteq\rangle$ and

$\langle \mathcal{P}(T), \subseteq \rangle$:

$$X^{\triangleright} = \{ b \in T \mid (\forall x \in X)(x\ N\ b) \},$$
$$Y^{\triangleleft} = \{ a \in M \mid (\forall y \in Y)(a\ N\ y) \}.$$

Lemma 2.3 (see e.g. [5, 8])**.** *The maps* $^{\triangleleft}$ *and* $^{\triangleright}$ *have the following properties:*

- $X \subseteq Y$ *implies* $Y^{\triangleright} \subseteq X^{\triangleright}$ *for* $X, Y \subseteq M$*.*
- $X \subseteq Y$ *implies* $Y^{\triangleleft} \subseteq X^{\triangleleft}$ *for* $X, Y \subseteq T$*.*
- $\emptyset^{\triangleleft} = M$ *and* $\emptyset^{\triangleright} = T$*.*
- $X^{\triangleright\triangleleft\triangleright} = X^{\triangleright}$ *and* $Y^{\triangleleft\triangleright\triangleleft} = Y^{\triangleleft}$ *for* $X \subseteq M$ *and* $Y \subseteq T$*.*
- *The map* $\gamma_N \colon \mathcal{P}(M) \to \mathcal{P}(M)$ *defined by* $\gamma_N(X) = X^{\triangleright\triangleleft}$ *is a closure operator on* $\langle \mathcal{P}(M), \subseteq \rangle$*.*
- *The collection* $\{ \{b\}^{\triangleleft} \mid b \in T \}$ *forms a basis for* γ_N*.*

Let $x_1, \ldots, x_n \in M$. To shorten the notation, we will write $\gamma_N\{x_1, \ldots, x_n\}$ instead of $\gamma_N(\{x_1, \ldots, x_n\})$.

Assume that we have a closure operator γ on the powerset monoid $\mathcal{P}(\mathbf{M})$ described in Example 2.1. If γ satisfies $\gamma(\gamma(X) \cdot \gamma(Y)) = \gamma(X \cdot Y)$ for all $X, Y \subseteq M$ then γ is called a *nucleus*. In this case one can define a residuated lattice on γ-closed sets. The resulting algebra $\mathcal{P}(\mathbf{M})_\gamma = \langle \mathcal{P}(M)_\gamma, \cap, \cup_\gamma, \cdot_\gamma, \backslash, /, \gamma\{1\} \rangle$, where $X \cup_\gamma Y = \gamma(X \cup Y)$ and $X \cdot_\gamma Y = \gamma(X \cdot Y)$, is a residuated lattice (see e.g. [8]).

We have mentioned above that every binary relation $N \subseteq M \times T$ induces a closure operator γ_N on $\langle \mathcal{P}(M), \subseteq \rangle$. The following definition gives a sufficient condition on N for γ_N to be in addition a nucleus.

Definition 2.2 ([7])**.** A residuated frame is a two-sorted structure $\mathbf{W} = \langle \mathbf{M}, T, N \rangle$ where $\mathbf{M} = \langle M, \cdot, 1 \rangle$ is a monoid, T is a set, and $N \subseteq M \times T$ is a nuclear relation, i.e. there exist operations $\backslash\!\!\backslash \colon M \times T \to T$ and $/\!\!/ \colon T \times M \to T$ such that

$$x \cdot y\ N\ z \quad \text{iff} \quad y\ N\ x \backslash\!\!\backslash z \quad \text{iff} \quad x\ N\ z /\!\!/ y.$$

Given a residuated frame $\mathbf{W} = \langle \mathbf{M}, T, N \rangle$, the induced closure operator γ_N is a nucleus on the powerset monoid $\mathcal{P}(\mathbf{M})$ (see [7]) indeed. Thus one can define a dual algebra $\mathbf{W}^+$ of the residuated frame $\mathbf{W}$ by letting $\mathbf{W}^+$ to be the residuated lattice $\mathcal{P}(\mathbf{M})_{\gamma_N}$.

Now we present an example of a residuated frame associated with a language L over an alphabet Σ and its dual algebra. This example will be of use later. Given an alphabet Σ, the set of all words (resp. non-empty words) over Σ is denoted Σ^* (resp. Σ^+). Recall that Σ^* together with the concatenation of words and the empty word ε forms the free Σ-generated monoid.

Example 2.2. Let Σ be an alphabet and $L \subseteq \Sigma^*$ a language. Consider a structure $\mathbf{W}_L = \langle \Sigma^*, \Sigma^* \times \Sigma^*, N \rangle$ where the binary relation $N \subseteq \Sigma^* \times (\Sigma^* \times \Sigma^*)$ is defined by

$$x\ N\ \langle u, v \rangle \quad \text{iff} \quad uxv \in L.$$

It follows that N is nuclear, since for all $x, y, u, v \in \Sigma^*$ we have

$$xy\ N\ \langle u, v \rangle \quad \text{iff} \quad y\ N\ \langle ux, v \rangle \quad \text{iff} \quad x\ N\ \langle u, yv \rangle \quad \text{iff} \quad uxyv \in L.$$

Consequently, $\mathbf{W}_L$ forms a residuated frame and the dual algebra

$$\mathbf{W}_L^+ = \langle W_L^+, \cap, \cup_{\gamma_N}, \cdot_{\gamma_N}, \backslash, /, \gamma_N\{\varepsilon\} \rangle$$

is a residuated lattice where $W_L^+ = \mathcal{P}(\Sigma^*)_{\gamma_N}$, $X \cup_{\gamma_N} Y = \gamma_N(X \cup Y)$, and $X \cdot_{\gamma_N} Y = \gamma_N(X \cdot Y)$.

In what follows, we assume that $\Sigma \subseteq \mathit{Var}$. Hence $\Sigma^* \subseteq \mathit{Fm}$ if we identify a word $a_1 a_2 \dots a_n \in \Sigma^*$ with the formula $a_1 \cdot a_2 \cdot \dots \cdot a_n$ (the fusion of atoms $a_1, \dots, a_n$).

Lemma 2.4. *Let $e\colon \mathit{Fm} \to W^+$ be a $\mathbf{W}_L^+$-evaluation, $a_1, \dots, a_n \in \Sigma$, and $w = a_1 a_2 \dots a_n$. Assume that $e(a_i) = \gamma_N(X_i)$ and $X_i \subseteq \Sigma^*$ for $i = 1, \dots, n$. Then $e(w) = \gamma_N(X_1 \cdot X_2 \cdot \dots \cdot X_n)$. In particular, if $X_i = \{a_i\}$ for $i = 1, \dots, n$ then $e(w) = \gamma_N\{w\}$.*

Proof. By the definition of a nucleus we have

$$\gamma_N(X) \cdot_{\gamma_N} \gamma_N(Y) = \gamma_N(\gamma_N(X) \cdot \gamma_N(Y)) = \gamma_N(X \cdot Y).$$

This can be easily generalized for arbitrarily many subsets $X_1, \dots, X_n \subseteq \Sigma^*$ and hence

$$\gamma_N(X_1) \cdot_{\gamma_N} \dots \cdot_{\gamma_N} \gamma_N(X_n) = \gamma_N(X_1 \cdots X_n).$$

Therefore the lemma follows since

$$e(w) = e(a_1) \cdot_{\gamma_N} \dots \cdot_{\gamma_N} e(a_n) = \gamma_N(X_1) \cdot_{\gamma_N} \dots \cdot_{\gamma_N} \gamma_N(X_n) = \gamma_N(X_1 \cdots X_n).$$

$\square$

Certainly, we are interested in languages L such that $\mathbf{W}_L^+$ is square increasing. We say that a language L over an alphabet Σ is closed under the contraction rule if $uxxv \in L$ implies $uxv \in L$ for all $u, x, v \in \Sigma^*$.

Lemma 2.5. *If $L \subseteq \Sigma^*$ is closed under the contraction rule then $\mathbf{W}_L^+ \in \mathcal{RL}_c$.*

Proof. Let $X \in W_L^+$, i.e. $X = \gamma_N(X)$. It suffices to show that $(X \cdot X)^{\triangleright} \subseteq X^{\triangleright}$, for if this is proved, Lemma 2.3 gives

$$X = X^{\triangleright\triangleleft} \subseteq (X \cdot X)^{\triangleright\triangleleft} = X \cdot_{\gamma_N} X.$$

Assume that $\langle u, v\rangle \in (X \cdot X)^{\triangleright}$ and $x \in X$. Hence $xx \in X \cdot X$ and so $uxxv \in L$. Since L is closed under the contraction rule, we have $uxv \in L$. Thus $\langle u, v\rangle \in X^{\triangleright}$. $\square$

2.2. **Regular languages.** In what follows, we will need to encode regular languages closed under the contraction rule into the equational theory of $\mathcal{RL}_c$. In this section we will show how to do it. It also affords a good illustration of how to use residuated frames in order to prove a completeness of an encoding.

Given an alphabet Σ, the set of all regular languages over Σ is denoted $\mathrm{Reg}(\Sigma)$. Recall that every regular language can be captured by a right-linear context-free grammar (see [11]). Let $G = \langle V, \Sigma, P, S\rangle$ be a right-linear context-free grammar with a finite set of variables V (non-terminals), a finite set of terminals Σ, a start variable S, and a finite set of production rules P of the form $A \to wB$ or $A \to w$ for some variables $A, B \in V$ and $w \in \Sigma^*$. The derivation relation $\to_G^*$ of G is defined in the usual way (see e.g. [11]). For every non-terminal $A \in V$ we define its language $L(A) = \{\, w \in \Sigma^* \mid A \to_G^* w \,\}$. In particular, $L(S)$ is the language generated by G. Until further notice, we assume that $V \cup \Sigma \subseteq \mathit{Var}$.

We define a finite set of formulae

$$\Delta_G = \{\, wB\backslash A \mid A \to wB \in P \,\} \cup \{\, w\backslash A \mid A \to w \in P \,\}.$$

Then we define a formula δ_G as the meet of 1 and all the formulae in Δ_G, i.e. $\delta_G = 1 \wedge \bigwedge \Delta_G$.

Now we can encode the membership of a word $w \in \Sigma^*$ in the regular language $L = L(S)$ generated by G via the identity $w\delta_G \leq S$. The following lemma shows that this encoding is sound.

Lemma 2.6. *Let $G = \langle V, \Sigma, P, S\rangle$ be a right-linear context-free grammar generating a regular language L and $w \in (V \cup \Sigma)^*$. If $S \to_G^* w$ then $w\delta_G \leq S$ holds in $\mathcal{RL}_c$. In particular, if $w \in L = L(S)$ then $w\delta_G \leq S$ holds in $\mathcal{RL}_c$.*

Proof. The claim is proved by induction on the number of steps in the derivation of w using the grammar G. Clearly, $S\delta_G \leq S$ holds in $\mathcal{RL}_c$ since $\delta_G \leq 1$. Assume that w is derived by a production rule $A \to uB \in P$, i.e. $w = w'uB$. Hence $uB\backslash A \in \Delta_G$ and so $\delta_G \leq uB\backslash A$. By the induction hypothesis, we know that $w'A\delta_G \leq S$ holds in $\mathcal{RL}_c$. It follows that

$$w\delta_G \leq w\delta_G^2 \leq w(uB\backslash A)\delta_G = w'uB(uB\backslash A)\delta_G \leq w'A\delta_G \leq S.$$

The case for a production rule of the form $A \to u$ is completely analogous. □

We prove the completeness of our encoding via the residuated frame $\mathbf{W}_L$ (see Example 2.2). We start with a general lemma.

Lemma 2.7. *Let $G = \langle V, \Sigma, P, S\rangle$ be a right-linear context-free grammar and $\mathbf{W} = \langle \Sigma^*, T, N\rangle$ a residuated frame. Given a $\mathbf{W}^+$-evaluation $e\colon Fm \to W^+$ suppose that $e(a) = \gamma_N\{a\}$ for $a \in \Sigma$ and $e(A) = \gamma_N(L(A))$ for $A \in V$. Then $\varepsilon \in e(\delta_G)$.*

Proof. We have to show that $\varepsilon \in e(\delta_G) = e(1) \cap \bigcap_{\varphi\in\Delta_G} e(\varphi)$. Since $\varepsilon \in \gamma_N\{\varepsilon\} = e(1)$, it suffices to show that for every $\varphi \in \Delta_G$ we have $\varepsilon \in e(\varphi)$. In other words, we need to show that $e(wB) \subseteq e(A)$ (resp. $e(w) \subseteq e(A)$) if $\varphi = wB\backslash A$ (resp. $\varphi = w\backslash A$).

Assume that $\varphi = wB\backslash A$ (the proof for $\varphi = w\backslash A$ is analogous). Hence the production rule $A \to wB$ belongs to P. Let $x \in L(B)$. Hence $A \to_G wB \to_G^* wx$, i.e. $wx \in L(A)$. Consequently, we have $\{w\} \cdot L(B) \subseteq L(A)$. Thus Lemma 2.4 implies

$$e(wB) = \gamma_N(\{w\} \cdot L(B)) \subseteq \gamma_N(L(A)) = e(A).$$

□

Assume that the regular language L generated by G is closed under the contraction rule and $w\delta_G \leq S$ holds in $\mathcal{RL}_c$. Hence $\mathbf{W}_L^+$ belongs to $\mathcal{RL}_c$ by Lemma 2.5. Thus $w\delta_G \leq S$ holds in $\mathbf{W}_L^+$. Consider the evaluation from Lemma 2.7. It follows that $e(w\delta_G) = \gamma_N(e(w)\cdot e(\delta_G)) \subseteq e(S) = \gamma_N\{L\}$. Since $\varepsilon \in e(\delta_G)$ by Lemma 2.7 and $w \in \gamma_N\{w\} = e(w)$ by Lemma 2.4, we obtain $w \in e(w) \cdot e(\delta_G) \subseteq e(w\delta_G) \subseteq \gamma_N\{L\}$. In order to show that $w \in L$, it suffices to show that L is γ_N-closed. To see this observe that $L = \{\langle\varepsilon, \varepsilon\rangle\}^{\lhd}$. Thus L is a member of the basis for γ_N (see Lemma 2.3).

Theorem 2.8. *Let L be a regular language closed under the contraction rule, $G = \langle V, \Sigma, P, S\rangle$ its generating right-linear context-free grammar, and $\delta_G = 1 \wedge \bigwedge \Delta_G$. Then $w \in L$ iff $w\delta_G \leq S$ holds in $\mathcal{RL}_c$.*

3. SRSs and atomic conditional SRSs

A *string rewriting system* (shortly SRS) is a tuple $\langle \Sigma, R\rangle$, where Σ is an alphabet and $R \subseteq \Sigma^* \times \Sigma^*$ is a binary relation. A member $\langle x, y\rangle$ of R is called a (rewriting) rule and we write it $x \rightsquigarrow y$.

A single-step reduction relation $\rightsquigarrow_R \subseteq \Sigma^* \times \Sigma^*$ is defined for any $w, w_1 \in \Sigma^*$ as $w \rightsquigarrow_R w_1$ iff there are words $x, y, u, v \in \Sigma^*$ such that $w = uxv$, $w_1 = uyv$, and $x \rightsquigarrow y \in R$. A reduction relation $\rightsquigarrow_R^*$ is the reflexive transitive closure of $\rightsquigarrow_R$. For further details see e.g. [1].

Given a word $w_0 \in \Sigma^*$, we define a language corresponding to $\langle \Sigma, R\rangle$ and w_0 by

$$L(w_0) = \{\, w \in \Sigma^* \mid w \rightsquigarrow_R^* w_0 \,\}.$$

The problem to decide whether a word $w \in \Sigma^*$ belongs to $L(w_0)$ is sometimes called the reachability problem (for a fixed word w_0).

It is easy to construct an SRSs $\langle \Sigma, R\rangle$ and $w_0 \in \Sigma^+$ such that $L(w_0)$ is undecidable. A common way how to obtain such a rewriting system is to encode a Minsky machine (two-counter machine) with an undecidable set of accepting configurations. These machines have a finite set of states and therefore their configuration can be completely described by a triplet $\langle i, m, n\rangle$ of natural numbers, which says that the machine is in the state i and counters have values m and n. A possible way how to encode such a triplet by a word is

$$Aa^m q_i a^n B,$$

where A, B are stoppers and a^k is the sequence of k letters a. One can also capture operations of Minsky machines by rewriting rules. It follows that the language $L(Aq_0B)$ is undecidable—a Minsky machine accepts a configuration $\langle i, m, n\rangle$ if its computation ends in $\langle 0, 0, 0\rangle$. However, in our case the problem is that $L(Aq_0B)$ is not closed under the contraction rule. Therefore such a straightforward representation of counters is impossible.

Nevertheless, we can represent counters by square-free words, i.e. do not contain uu as a subword. It is well known (see e.g. [21]) that if we have a morphism over the alphabet $\{a, b, c\}$ defined by

$$h(a) = abc \qquad h(b) = ac \qquad h(c) = b$$

then $h^m(a)$ is square free for any m. Hence we can represent a state of our machine by

$$Ah^m(a)Bq_iCh^n(a)D.$$

In this way we can obtain a rewriting system such that the language $L(AaBq_0CaD)$ is undecidable and consists only of square-free words and therefore it is closed under the contraction rule. This coding is inspired by [15, Section 7.2.5] and the complete construction is described in [12, Section 4], where the word problem for $\mathcal{RL}_c$ and therefore the deducibility problem for $\mathbf{FL_c^+}$ are proved to be undecidable.

Theorem 3.1 ([12])**.** *There is an SRS $\langle \Sigma, R\rangle$ and $w_0 \in \Sigma^+$ such that $L(w_0)$ is undecidable. In addition, $L(w_0)$ consists only of square-free words, i.e. $L(w_0)$ is closed under the contraction rule. Moreover, the rules contain only square-free words.*

It is worth pointing out that R from the previous theorem contains only non-empty words, i.e. $R \subseteq \Sigma^+ \times \Sigma^+$, and hence if $w \rightsquigarrow_R^* w_0$ then $w \in \Sigma^+$ since $w_0 \in \Sigma^+$. Clearly, empty words play no essential role in this SRS,[2] the fact to be used later.

In this paper we will present an encoding of the reduction relation $\rightsquigarrow_R^*$ for such a rewriting system into the equational theory of $\mathcal{RL}_c$.

3.1. **A naïve way of encoding.** Theorem 3.1 gives us an SRS $\langle \Sigma, R\rangle$ and $w_0 \in \Sigma^+$ such that it is undecidable whether $w \rightsquigarrow_R^* w_0$ for $w \in \Sigma^+$. Algebraically we would like to encode this problem as the validity of $w \leq w_0$ modulo the given set of rules R. The problem is how to represent the set of rules R. Now we are going to present a naïve way how to do that. Although it does not work, we will elaborate on it later on.

The most natural way is to represent a rule $x \rightsquigarrow y \in R$ as an implication, e.g. $x\backslash y$. The idea is as follows. Assume we have $uxv, uyv \in \Sigma^*$. We know $x(x\backslash y) \leq y$ (see Fact 2.1) and

[2] For these reasons even the definition of SRSs in [12] allows only non-empty words.

hence $ux(x\backslash y)v \leq uyv$. Moreover, as we have contraction, it holds that

$$ux(x\backslash y)v \leq ux(x\backslash y)(x\backslash y)v \leq uy(x\backslash y)v.$$

This shows how to represent a rewriting rule, but we can easily generalize this to the whole set R using meet. Let $\theta = \bigwedge_{x \rightsquigarrow y \in R}(x\backslash y)$ then in the previous example we have

$$ux\theta v \leq ux\theta\theta v \leq ux(x\backslash y)\theta v \leq uy\theta v.$$

However, such a straightforward representation fails. Assume also $z \rightsquigarrow xv \in R$. We have $uz \rightsquigarrow_R uxv \rightsquigarrow_R uyv$. Hence we would like to show that $uz\theta \leq ux\theta v \leq uy\theta v$, but the previous technique does not work. It is not enough that $z\backslash xv$ is in θ, because we would need $z\backslash x\theta v$ to be in θ, which is obviously impossible since it cannot contain itself.

Obviously, this is an essential problem. If all $x \rightsquigarrow y \in R$ were such that y is only a letter, i.e. $y \in \Sigma$, then this would work, but for obvious reasons there is no such SRS satisfying Theorem 3.1. However, it is possible to define a modification of SRSs (Section 3.2) such that the naïve way of encoding will be applicable on these modified systems (Section 4).

3.2. **Conditional string rewriting systems.** To overcome the problem with the naïve encoding, we introduce a certain modification of string rewriting systems, which we call conditional SRSs. A *conditional string rewriting system* (or CSRS) is a tuple $\langle \Sigma, R \rangle$, where Σ is an alphabet and $R \subseteq \Sigma^* \times \Sigma^* \times \mathrm{Reg}(\Sigma) \times \mathrm{Reg}(\Sigma)$ is a relation. A member $\langle x, y, L_\ell, L_r \rangle$ of R consists of a rewriting rule $x \rightsquigarrow y$ and two regular languages L_ℓ, L_r and expresses the fact that the rule $x \rightsquigarrow y$ can be used only in a context restricted by the languages L_ℓ, L_r. We denote the tuple $\langle x, y, L_\ell, L_r \rangle$ more suggestively $\langle x \rightsquigarrow y, L_\ell, L_r \rangle$. A single-step reduction relation $\rightsquigarrow_R \subseteq \Sigma^* \times \Sigma^*$ is defined for any $w, w_1 \in \Sigma^*$ by

$w \rightsquigarrow_R w_1$ iff there are a rule $\langle x \rightsquigarrow y, L_\ell, L_r \rangle \in R$, and words $u \in L_\ell$ and $v \in L_r$ such that $w = uxv$ and $w_1 = uyv$.

A reduction relation $\rightsquigarrow_R^*$ is the reflexive transitive closure of $\rightsquigarrow_R$.

Note that similar rewriting systems were considered in the literature. For instance, Chottin in [2] defined so-called controlled string rewriting systems where only left contexts are restricted by regular languages.

A CSRS is said to be *atomic* if all its rules have atomic right-hand sides, i.e. if for every $\langle x \rightsquigarrow y, L_\ell, L_r \rangle$ in R we have $y \in \Sigma$.

In the rest of this section we are going to show that every SRS $\langle \Sigma, R \rangle$, which has $R \subseteq \Sigma^* \times \Sigma^+$,[3] can be simulated by an atomic CSRS. More precisely, assume that we have another two copies of Σ denoted $\Sigma' = \{ a' \mid a \in \Sigma \}$ and $\Sigma'' = \{ a'' \mid a \in \Sigma \}$ such that Σ, Σ', and Σ'' are disjoint. We will prove that there is an atomic CSRS $\langle \Sigma \cup \Sigma' \cup \Sigma'', R' \rangle$ such that for every $w, w_0 \in \Sigma^*$ we have $w \rightsquigarrow_R^* w_0$ iff $w \rightsquigarrow_{R'}^* w_0$.

Clearly, every rule $x \rightsquigarrow a$ in R with an atomic right-hand side can be simulated by the atomic conditional rule $\langle x \rightsquigarrow a, \Sigma^*, \Sigma^* \rangle$. Every non-atomic rule $x \rightsquigarrow a_1 \dots a_n$ from R, where

[3] This assumption is useful for simplifying the construction since the SRS from Theorem 3.1 satisfies it. However, we could extend the construction even to rules of the form $x \rightsquigarrow \varepsilon$. Roughly speaking, it would suffice to allow empty words in the definition of atomic CSRSs and handle such rules similarly to atomic rules.

$n \geq 2$ and $a_1, \ldots, a_n \in \Sigma$, is simulated by the following atomic conditional rules:

$$\langle \varepsilon \rightsquigarrow a_i'', \Sigma^*(\Sigma'')^*, \Sigma^* \rangle \qquad \text{for } i \in \{2, \ldots, n\}, \tag{3}$$

$$\langle x \rightsquigarrow a_1', \Sigma^*, a_2'' \ldots a_n'' \Sigma^* \rangle, \tag{4}$$

$$\langle a_i'' \rightsquigarrow a_i, \Sigma^* \Sigma' (\Sigma'')^*, \Sigma^* \rangle \qquad \text{for } i \in \{2, \ldots, n\}, \tag{5}$$

$$\langle a_1' \rightsquigarrow a_1, \Sigma^*, \Sigma^* \rangle. \tag{6}$$

Lemma 3.2. *Let $w, w_0 \in \Sigma^*$. Then $w \rightsquigarrow_R^* w_0$ implies $w \rightsquigarrow_{R'}^* w_0$.*

Proof. By induction on the length of derivation. The simulation of a rule $x \rightsquigarrow a$ with an atomic right-hand side is obvious. Let $x \rightsquigarrow a_1 \ldots a_n$ be a rule in R having a non-atomic right-hand side. The rewriting step $uxv \rightsquigarrow_R ua_1 \ldots a_n v$ for $u, v \in \Sigma^*$ is simulated as follows:

$$\begin{aligned}
uxv &\rightsquigarrow_{R'} uxa_2''v && \text{by (3)} \\
&\rightsquigarrow_{R'}^* uxa_2'' \ldots a_n''v && \text{by (3)} \\
&\rightsquigarrow_{R'} ua_1'a_2'' \ldots a_n''v && \text{by (4)} \\
&\rightsquigarrow_{R'}^* ua_1'a_2 \ldots a_n v && \text{by (5)} \\
&\rightsquigarrow_{R'} ua_1a_2 \ldots a_n v && \text{by (6).}
\end{aligned}$$

□

For the converse direction we need to interpret the auxiliary words containing symbols from Σ' and Σ'' back in Σ^*. For this purpose we define two monoid homomorphisms

$$h_1 \colon (\Sigma \cup \Sigma' \cup \Sigma'')^* \to \Sigma^* \quad \text{and} \quad h_2 \colon (\Sigma \cup \Sigma'')^* \to \Sigma^*$$

respectively by

$$h_1(a) = h_1(a') = h_1(a'') = a \quad \text{and} \quad h_2(a) = a,\ h_2(a'') = \varepsilon$$

for all $a \in \Sigma$.

Since the domains of h_1 and h_2 are the free monoids, the above definitions extend uniquely to the whole domains. Then we merge the above homomorphisms together and define a mapping $h \colon (\Sigma \cup \Sigma' \cup \Sigma'')^* \to \Sigma^*$ by

$$h(w) = \begin{cases} h_2(w) & \text{if } w \in (\Sigma \cup \Sigma'')^*, \\ h_1(w) & \text{otherwise, i.e. } w \text{ contains a letter from } \Sigma'. \end{cases}$$

Note that $h(w) = h_2(w) = w$ for $w \in \Sigma^*$.

Lemma 3.3. *If $w \rightsquigarrow_{R'}^* w_0$ then $h(w) \rightsquigarrow_R^* h(w_0)$ for $w, w_0 \in (\Sigma \cup \Sigma' \cup \Sigma'')^*$. In particular, $w \rightsquigarrow_{R'}^* w_0$ implies $w \rightsquigarrow_R^* w_0$ for $w, w_0 \in \Sigma^*$.*

Proof. By induction on the length of derivation. Assume that $w \rightsquigarrow_{R'} w_1 \rightsquigarrow_{R'}^* w_0$. By the induction hypothesis, we have $h(w_1) \rightsquigarrow_R^* h(w_0)$. If $w = uxv$, $w_1 = uav$ for $u, v, x \in \Sigma^*$, $a \in \Sigma$, and $\langle x \rightsquigarrow a, \Sigma^*, \Sigma^* \rangle \in R'$ then $x \rightsquigarrow a \in R$ and the lemma holds trivially. Assume that the rule from R' used in $w \rightsquigarrow_{R'} w_1$ is among the rules (3)–(6) corresponding to a rule $x \rightsquigarrow a_1 \ldots a_n$ in R. The proof splits into two cases.

First, suppose that $w \in (\Sigma \cup \Sigma'')^*$. Hence $h(w) = h_2(w)$. Note that only the rule (3) or (4) can be applied to w. If (3) is applied then $w = uv$ and $w_1 = ua_i''v$ for some $u \in \Sigma^*(\Sigma'')^*$, $v \in \Sigma^*$, and $i \in \{2, \ldots, n\}$. Thus $w_1 \in (\Sigma \cup \Sigma'')^*$ and we have

$$h(w_1) = h_2(w_1) = h_2(ua_i''v) = h_2(uv) = h(w).$$

Therefore $h(w) \rightsquigarrow_R^* h(w_1)$ by reflexivity.

If (4) is applied then $w = uxa_2'' \dots a_n''v$ and $w_1 = ua_1'a_2'' \dots a_n''v$ for some $u, v, x \in \Sigma^*$. Hence

$$h(w) = h_2(w) = h_2(uxa_2'' \dots a_n''v) = uxv$$

and

$$h(w_1) = h_1(ua_1'a_2'' \dots a_n''v) = ua_1a_2 \dots a_nv.$$

Consequently, we have $h(w) = uxv \rightsquigarrow_R ua_1a_2 \dots a_nv = h(w_1)$.

Second, suppose that $w \notin (\Sigma \cup \Sigma'')^*$, i.e. w contains a letter $a' \in \Sigma'$. Hence $h(w) = h_1(w)$ and only the rule (5) or (6) can be applied to w. If (5) is applied then $w = ua'z''a_i''v$ and $w_1 = ua'z''a_iv$ for some $u, v \in \Sigma^*$, $a' \in \Sigma'$, $z'' \in (\Sigma'')^*$, and $i \in \{2, \dots, n\}$. This gives

$$h(w) = h_1(w) = h_1(ua'z''a_i''v) = h_1(ua'z'')a_ih_1(v) = h_1(ua'z''a_iv) = h(w_1)$$

and so $h(w) \rightsquigarrow_R^* h(w_1)$.

Finally, if (6) is applied then $w = ua_1'v$ and $w_1 = ua_1v$ for some $u, v \in \Sigma^*$. We thus get $h(w) = h_1(w) = ua_1v = h_2(w_1) = h(w_1)$ and so $h(w) \rightsquigarrow_R^* h(w_1)$. □

Assume that the language $L(w_0)$ corresponding to the original SRS $\langle \Sigma, R \rangle$ consists of square-free words only. The next lemma shows that the language

$$L'(w_0) = \{\, w \in (\Sigma \cup \Sigma' \cup \Sigma'')^* \mid w \rightsquigarrow_{R'}^* w_0 \,\}$$

associated with the atomic CSRS $\langle \Sigma \cup \Sigma' \cup \Sigma'', R' \rangle$ also contains only square-free words.

Lemma 3.4. *The language $L'(w_0) \subseteq (\Sigma \cup \Sigma' \cup \Sigma'')^*$ contains only square-free words.*

Proof. Let $w \in L'(w_0)$. We prove the lemma by induction on the length of derivation. The base case is easy since $w_0 \in L(w_0)$ is a square-free word. Suppose that $w \rightsquigarrow_{R'} w_1 \rightsquigarrow_{R'}^* w_0$. By the induction hypothesis, w_1 is square free. We distinguish five cases according to the rule used in $w \rightsquigarrow_{R'} w_1$. Before that note the following general fact. Since $w \in L'(w_0)$, we have $h(w) \rightsquigarrow_R^* h(w_0) = w_0$ by Lemma 3.3. Hence $h(w) \in L(w_0)$.

If a rule $\langle x \rightsquigarrow a, \Sigma^*, \Sigma^* \rangle \in R'$ corresponding to an atomic rule $x \rightsquigarrow a \in R$ is used then $w \in \Sigma^*$. Consequently, $w = h(w) \in L(w_0)$ and $L(w_0)$ contains only square-free words.

Assume that the rule from R' used in $w \rightsquigarrow_{R'} w_1$ is among the rules (3)–(6) corresponding to a rule $x \rightsquigarrow a_1 \dots a_n$ in R.

If (3) is used then $w = uu''v$ and $w_1 = uu''a_i''v$ for $u, v \in \Sigma^*$ and $u'' \in (\Sigma'')^*$. Since u'' is a subword of w_1, it follows that u'' is square free. If $u'' \neq \varepsilon$ then w is square free, because otherwise u or v would contain a square, which contradicts w_1 being square free. If $w = uv$ then $w = h(w) \in L(w_0)$ and so it is square free.

If (4) is used then $w = uxa_2'' \dots a_n''v$ and $w_1 = ua_1'a_2'' \dots a_n''v$ for some $u, v \in \Sigma^*$. Since w_1 is square free, the same holds for u and v. If w contained a square then it would have to be a subword of ux. Since $h(w) = h_2(w) = uxv \in L(w_0)$, uxv is square free. Thus ux is square free as well, a contradiction.

If (5) or (6) is used then w contains a letter from Σ'. Thus $h(w) = h_1(w)$. Suppose that $w = uzzv$ for some $u, z, v \in (\Sigma \cup \Sigma' \cup \Sigma'')^*$. Hence

$$h(w) = h_1(uzzv) = h_1(u)h_1(z)h_1(z)h_1(v) \in L(w_0).$$

Since $L(w_0)$ contains only square-free words, we have $h_1(z) = \varepsilon$. Thus $z = \varepsilon$ since $h_1^{-1}(\varepsilon) = \{\varepsilon\}$. Consequently, $w = uzzv = uv$. □

It is easy to see that the conditional languages Σ^*, $\Sigma^*(\Sigma'')^*$, and $\Sigma^*\Sigma'(\Sigma'')^*$ are closed under the contraction rule. Also the last conditional language $a''_2 \dots a''_n\Sigma^*$ is closed under the contraction rule because the right-hand sides of all rules in R are square free (see Theorem 3.1). Summarizing, we have the following theorem.

Theorem 3.5. *There is an atomic CSRS $\langle \Sigma, R\rangle$ and $w_0 \in \Sigma^+$ such that the corresponding language $L(w_0)$ is undecidable and consists only of square-free words. Moreover, the conditional regular languages are closed under the contraction rule.*

4. Atomic conditional SRSs and $\mathcal{RL}_c$

In this section we show how to encode an atomic CSRS into the equational theory of $\mathcal{RL}_c$. Let $\langle \Sigma, R\rangle$ be the atomic CSRS from Theorem 3.5 and $w_0 \in \Sigma^+$ such that the language $L(w_0) = \{ w \in \Sigma^+ \mid w \rightsquigarrow^*_R w_0 \}$ consists only of square-free words (i.e. it is closed under the contraction rule) and is undecidable. Also conditional languages of every rule in R are closed under the contraction rule.

First, we describe the conditional contexts in our atomic CSRS by a right-linear context-free grammar. We can index the members of R by an index set I, i.e. $R = \{ R_i \mid i \in I \text{ and } R_i \text{ is a rule} \}$. Define an extended alphabet $\Sigma_e = \Sigma \cup \{ r_i \mid i \in I \}$, where r_i are fresh variables. Of course, we assume that $\Sigma_e \subseteq \mathit{Var}$. For every rule $R_i = \langle x \rightsquigarrow a, L_\ell, L_r\rangle$, where $x \in \Sigma^*$ and $a \in \Sigma$, define a regular language $L_i = L_\ell r_i L_r$. Note that the languages L_i are pairwise disjoint due to the pairwise different symbols r_i and closed under the contraction rule. Finally, we define a regular language $L_{\mathsf{Aux}} = \bigcup_{i\in I} L_i$, which is clearly closed under the contraction rule. Since L_{Aux} is regular, there is a right-linear context-free grammar G generating L_{Aux}. Consider the formula $\delta_G = 1 \wedge \bigwedge \Delta_G$ such that $w\delta_G \leq S$ holds in $\mathcal{RL}_c$ iff w belongs to L_{Aux} (see Theorem 2.8), which means $w = ur_iv$ for some $i \in I$, $R_i = \langle x \rightsquigarrow a, L_\ell, L_r\rangle$, $u \in L_\ell$, and $v \in L_r$.

Second, we can combine this with the naïve way of encoding rules from Section 3.1. As we have an atomic CSRS, which means only letters can occur on the right-hand side of rewriting rules, the main obstacle disappeared, cf. Section 3.1. Moreover, we have shown how to describe the conditional contexts using the grammar G and hence δ_G. Now we can modify the definition of θ from Section 3.1 in the following way.

For every rule $R_i = \langle x \rightsquigarrow a, L_\ell, L_r\rangle$ in R, we define a formula $\theta_i = x\backslash(a \vee r_i)$. Furthermore, we extend this for all the rules by defining a formula $\theta = 1 \wedge \bigwedge_{i\in I} \theta_i$. Note that $\theta \leq \theta_i$ for all $i \in I$ and $\theta \leq 1$. Hence $\theta \leq \theta^2 \leq 1 \cdot \theta = \theta$.

Assume we have $uxv, uav \in \Sigma^*$ and $R_i = \langle x \rightsquigarrow a, L_\ell, L_r\rangle \in R$. It is now impossible to show $ux\theta v \leq ua\theta v$ as in Section 3.1, because θ_i contains r_i. This is by purpose—the conditionality of rewriting must be taken into account. Namely, ur_iv belongs to L_{Aux} only if $u \in L_\ell$ and $v \in L_r$. This gives us

$$ur_i\theta v\delta_G \leq ur_iv\delta_G \leq S \leq ua\theta v \vee S,$$

where $ur_iv\delta_G \leq S$ certifies that we rewrite in the correct context. Moreover, using $\delta_G \leq 1$ we know

$$ua\theta v\delta_G \leq ua\theta v \leq ua\theta v \vee S.$$

Hence we can combine these two things together using join and distributivity from Fact 2.1 and so

$$u(a \vee r_i)\theta v\delta_G = ua\theta v\delta_G \vee ur_i\theta v\delta_G \leq ua\theta v \vee S.$$

We are now in a position to simulate the rewriting step $x \rightsquigarrow a$ of our atomic CSRS using the formula $\theta_i = x\backslash(a \vee r_i)$. Since $ux(x\backslash(a \vee r_i))\theta v\delta_G \leq u(a \vee r_i)\theta v\delta_G$ by Fact 2.1, we obtain

$$ux\theta v\delta_G \leq ux\theta^2 v\delta_G \leq ux\theta_i\theta v\delta_G = ux(x\backslash(a \vee r_i))\theta v\delta_G \leq u(a \vee r_i)\theta v\delta_G \leq ua\theta v \vee S.$$

It is clear that we need the formula θ to be spread everywhere along a word if we want to simulate an arbitrary rewriting step. For this reason, given a non-empty word $w = a_1 \dots a_n$ such that all a_i are letters, we define $w^\theta = a_1\theta a_2\theta \dots a_n\theta$.[4] Observe that $w^\theta \leq w$, $w^\theta \leq w^\theta\theta$, and $(uv)^\theta = u^\theta v^\theta$ hold for all non-empty words u, v, and w.

The following lemma shows in full details that the outlined construction can be used to describe atomic CSRSs in the language of $\mathcal{RL}_c$.

Lemma 4.1 (Soundness)**.** *Let $w \in L(w_0)$. Then $w^\theta\delta_G \leq w_0 \vee S$.*

Proof. By induction on the length of derivation. The base case is obvious since $w_0^\theta \leq w_0$ and $\delta_G \leq 1$. Hence $w_0^\theta\delta_G \leq w_0 \leq w_0 \vee S$. Assume that $w \rightsquigarrow_R w_1$ by a rule $R_i = \langle x \rightsquigarrow a, L_\ell, L_r\rangle$ and $w_1 \rightsquigarrow_R^* w_0$. By the induction hypothesis, we have $w_1^\theta\delta_G \leq w_0 \vee S$. Furthermore, we have $w = uxv$ and $w_1 = uav$ such that $u \in L_\ell$ and $v \in L_r$. We know that

$$\begin{aligned} w^\theta = u^\theta x^\theta v^\theta \leq u^\theta x^\theta \theta v^\theta \leq u^\theta x\theta v^\theta \leq u^\theta x\theta^2 v^\theta \leq u^\theta x(x\backslash(a \vee r_i))\theta v^\theta \leq \\ \leq u^\theta(a \vee r_i)\theta v^\theta = u^\theta a\theta v^\theta \vee u^\theta r_i\theta v^\theta = (uav)^\theta \vee (ur_iv)^\theta \leq w_1^\theta \vee ur_iv. \end{aligned}$$

Observe that $ur_iv \in L_i$. Thus $ur_iv\delta_G \leq S$ by Theorem 2.8. Consequently,

$$w^\theta\delta_G \leq (w_1^\theta \vee ur_iv)\delta_G = w_1^\theta\delta_G \vee ur_iv\delta_G \leq w_0 \vee S.$$

□

It remains to prove the opposite direction. Consider the residuated frame $\mathbf{W}_{L(w_0)\cup L_{\mathsf{Aux}}} = \langle \Sigma_e^*, \Sigma_e^* \times \Sigma_e^*, N\rangle$ where

$$z\ N\ \langle u, v\rangle \quad \text{iff} \quad uzv \in L(w_0) \cup L_{\mathsf{Aux}}.$$

Hence $\mathbf{W}^+_{L(w_0)\cup L_{\mathsf{Aux}}}$ is a residuated lattice. Moreover, $\mathbf{W}^+_{L(w_0)\cup L_{\mathsf{Aux}}} \in \mathcal{RL}_c$ by Lemma 2.5 because $L(w_0) \cup L_{\mathsf{Aux}}$ is closed under the contraction rule by Theorem 3.5.

Assume that $w^\theta\delta_G \leq w_0 \vee S$ holds in $\mathcal{RL}_c$ for some $w \in \Sigma^+$. We want to show that $w \in L(w_0)$. Since the inequality $w^\theta\delta_G \leq w_0 \vee S$ holds, we have $e(w^\theta\delta_G) \subseteq e(w_0 \vee S)$ for every $\mathbf{W}^+_{L(w_0)\cup L_{\mathsf{Aux}}}$-evaluation $e\colon Fm \to W^+_{L(w_0)\cup L_{\mathsf{Aux}}}$. Let e be the evaluation defined in Lemma 2.7, i.e. $e(a) = \gamma_N\{a\}$ for every $a \in \Sigma_e$ and $e(A) = \gamma_N(L(A))$ for every non-terminal from the grammar G generating the language $L_{\mathsf{Aux}} = L(S)$. Therefore $e(u) = \gamma_N\{u\}$ for every word $u \in \Sigma^*$ by Lemma 2.4. Furthermore, we have

$$e(w_0 \vee S) = \gamma_N(\gamma_N\{w_0\} \cup \gamma_N(L(S))) = \gamma_N(\{w_0\} \cup L_{\mathsf{Aux}}).$$

Since $\langle \varepsilon, \varepsilon\rangle \in (\{w_0\} \cup L_{\mathsf{Aux}})^\triangleright$, we have by Lemma 2.3 that

$$\gamma_N(\{w_0\} \cup L_{\mathsf{Aux}}) \subseteq \{\langle \varepsilon, \varepsilon\rangle\}^\triangleleft = L(w_0) \cup L_{\mathsf{Aux}}.$$

Thus we know that the words in $e(w^\theta\delta_G) = \gamma_N(e(w^\theta) \cdot e(\delta_G))$ belong to $L(w_0) \cup L_{\mathsf{Aux}}$.

Lemma 4.2. *It holds $\varepsilon \in e(\theta_i)$ for all $i \in I$. Consequently, $\varepsilon \in e(\theta) = \gamma_N\{\varepsilon\} \cap \bigcap_{i\in I} e(\theta_i)$.*

[4]Note that for our atomic CSRS $\langle \Sigma, R\rangle$ from Theorem 3.5 there is no need to define ε^θ because ε never occurs if we start rewriting from a non-empty word $w \in \Sigma^+$. In general, this need not be the case, but even then we could ensure such behaviour by expressing $w \rightsquigarrow_R^* w_0$ equivalently as $bw \rightsquigarrow_R^* bw_0$, where b is a fresh letter.

Proof. Consider the formula $\theta_i = x\backslash(a \vee r_i)$ corresponding to a rule $R_i = \langle x \rightsquigarrow a, L_\ell, L_r\rangle \in R$. It suffices to check that

$$\gamma_N\{x\} = e(x) \subseteq e(a \vee r_i) = \gamma_N(\gamma_N\{a\} \cup \gamma_N\{r_i\}) = \gamma_N\{a, r_i\}.$$

Using the basis for γ_N (see Lemma 2.3) and (2), we have to show that $\{a, r_i\} \subseteq \{\langle u, v\rangle\}^{\triangleleft}$ implies $x \in \{\langle u, v\rangle\}^{\triangleleft}$. Assume that $uav, ur_iv \in L(w_0) \cup L_{\mathsf{Aux}}$. Hence $uav \in L(w_0)$ and $ur_iv \in L_i \subseteq L_{\mathsf{Aux}}$ which yield $u \in L_\ell$ and $v \in L_r$. Consequently, $uxv \rightsquigarrow_R uav \rightsquigarrow^*_R w_0$ and so $uxv \in L(w_0)$, i.e. $x \in \{\langle u, v\rangle\}^{\triangleleft}$. □

Since $\varepsilon \in e(\theta)$, we have $w \in e(w^\theta)$. Similarly, $\varepsilon \in e(\delta_G)$ by Lemma 2.7 and so $w \in e(w^\theta)e(\delta_G)$. Consequently, $w \in \gamma_N(e(w^\theta)e(\delta)) = e(w^\theta\delta_G) \subseteq L(w_0) \cup L_{\mathsf{Aux}}$. As $w \notin L_{\mathsf{Aux}}$ we have $w \in L(w_0)$.

Lemma 4.3 (Completeness)**.** *Assume that $w^\theta\delta_G \leq w_0 \vee S$ holds in $\mathcal{RL}_c$ for $w \in \Sigma^+$. Then $w \in L(w_0)$.*

Since the problem whether $w \in L(w_0)$ is undecidable, the set $\{\, \varphi \leq \psi \mid \varphi \leq \psi$ holds in $\mathcal{RL}_c \,\}$ is undecidable as well. Thus we obtain the following theorem.

Theorem 4.4. *The equational theory of $\mathcal{RL}_c$ is undecidable. Consequently, the set of formulae provable in $\mathbf{FL_c^+}$ is undecidable.*

5. Conclusions

Theorem 4.4 implies also the undecidability of the logic $\mathbf{FL_c}$ since $\mathbf{FL_c^+}$ is its positive fragment. Recall that $\mathbf{FL_c}$ is an expansion of $\mathbf{FL_c^+}$ where the language is expanded by a constant 0 (see [8]). The sequent calculus for $\mathbf{FL_c}$ can be obtained from the one shown in Definition 2.1 by adding the following axiom and rule:

$$(0\mathrm{L})\ \frac{}{0 \Rightarrow} \qquad\qquad (0\mathrm{R})\ \frac{\Gamma \Rightarrow}{\Gamma \Rightarrow 0}$$

Moreover, the notion of a sequent is modified a little bit by allowing on the right-hand side a stoup, i.e. a formula or the empty sequence. Accordingly, one has to modify the rules from Definition 2.1 in an obvious way. The logic $\mathbf{FL_{co}}$ is an extension of $\mathbf{FL_c}$ by the right weakening rule:

$$(\mathrm{o})\ \frac{\Gamma \Rightarrow}{\Gamma \Rightarrow \varphi}$$

The logic $\mathbf{FL_c}$ is sound and complete with respect to the variety of pointed square-increasing residuated lattices (also called $\mathbf{FL_c}$-algebras). An $\mathbf{FL_c}$-algebra $\mathbf{A} = \langle A, \wedge, \vee, \cdot, \backslash, /, 0, 1\rangle$ is an algebra such that $\langle A, \wedge, \vee, \cdot, \backslash, /, 1\rangle \in \mathcal{RL}_c$ and $0 \in A$. Similarly, the logic $\mathbf{FL_{co}}$ is sound and complete with respect to a subvariety of $\mathbf{FL_c}$-algebras axiomatized by the identity $0 \leq x$.

The logics $\mathbf{FL_c}$ and $\mathbf{FL_{co}}$ have a common fragment, namely $\mathbf{FL_c^+}$. This follows easily for $\mathbf{FL_c}$ because every square-increasing residuated lattice is a reduct of an $\mathbf{FL_c}$-algebra, it suffices to interpret 0 arbitrarily. Concerning $\mathbf{FL_{co}}$, it is sufficient to show that every square-increasing residuated lattice $\mathbf{A}$ is embeddable into an $\mathbf{FL_c}$-algebra where 0 is its bottom element. Since $\mathbf{A}$ need not have a minimum, we can first embed it into its Dedekind–MacNeille completion $\bar{\mathbf{A}}$. Since the Dedekind–MacNeille completion preserves the identity $x \leq x^2$, we have $\bar{\mathbf{A}} \in \mathcal{RL}_c$ (see [4]). Consequently, $\bar{\mathbf{A}}$ forms an $\mathbf{FL_c}$-algebra where $0 \leq x$ holds if we interpret 0 as the bottom element, it exists as the lattice reduct of $\bar{\mathbf{A}}$ is complete.

Theorem 5.1. *The set of formulae provable in* $\mathbf{FL_c}$ *(resp.* $\mathbf{FL_{co}}$*) is undecidable.*

5.1. **Used language.** In our constructions we have used almost complete language, but this is not necessary. The constant 1 has been used for simplicity and can be easily eliminated. We know that $x = 1\backslash x$ and in all the remaining cases (in δ_G and θ) we can replace 1 by the meet of all $p\backslash p$ for all atoms p occurring in our construction.

Similarly, we can get rid of all fusions. First, we can change even the original problem $w \rightsquigarrow_R^* w_0$ from Theorem 3.1 into an equivalent problem $w \rightsquigarrow_R^* p$, where p is a fresh atom, by adding the rewriting rule $w_0 \rightsquigarrow_R p$. Second, using the same construction as in the paper we obtain an identity. We easily get an equivalent identity that contains no fusion using $x_1 \cdot \dots \cdot x_n \leq y$ iff $x_n \leq x_{n-1}\backslash(\dots\backslash(x_1\backslash y)\dots)$ and $x_1 \cdot \dots \cdot x_m\backslash y = x_m\backslash(\dots\backslash(x_1\backslash y)\dots)$. Notice that in our case such a y can only be an atom or join of atoms.

It should be also noted that we could use / instead of \ changing the construction accordingly. Therefore, we can clearly formulate the whole construction in the language containing only an implication, join and meet. It does not matter whether as an identity in $\mathcal{RL}_c$ or sequent in $\mathbf{FL_c^+}$ (with or without empty left-hand side).

5.2. **Knotted axioms.** It should be clear that the construction can be easily adapted for logics having a weaker form of contraction $x^k \leq x^l$, $1 \leq k < l$. Basically one has to change the encoding by replacing w^θ with w^{θ^k}, where if $w = a_1 \dots a_n$ then $w^{\theta^k} = a_1\theta^k \dots a_n\theta^k$. Furthermore, the final inequality is changed to $w^{\theta^k}\delta_G^k \leq w_0 \vee S$.

In order to modify our proof, note that the identity $x \leq x^2$ is used only for θ and δ_G. Since $\theta \leq 1$ and $\delta_G \leq 1$, we obtain $\theta^k = \theta^{k+1}$ and $\delta_G^k = \delta_G^{k+1}$. If 1 is not in the language and we change θ and δ_G according to Section 5.1 then we still have $a\theta^k = a\theta^{k+1}$ and $a\delta_G^k = a\delta_G^{k+1}$ for every atom a occurring in $w^{\theta^k}\delta_G^k \leq w_0 \vee S$, which is sufficient to complete the proof.

5.3. **Deduction theorem.** From some point of view, one can understand our construction as a form of deduction theorem for a very limited fragment of formulae—a reachability problem for some rewriting systems is translated into provability in $\mathbf{FL_c^+}$. However, this suffices to get a full form of "algorithmic" deduction theorem, because we can easily obtain the following chain of reductions. Let φ be a formula and T a finite theory. First, the set of formulae provable in $\mathbf{FL_c}$ from T is recursively enumerable and hence there is a Minsky machine accepting an input (a suitable encoding of φ and T) iff φ is provable in $\mathbf{FL_c}$ from T. Second, this paper describes how to express such a decision problem in terms of provability in $\mathbf{FL_c^+}$. Moreover, all the steps in this chain are constructive and explicit.

Theorem 5.2. *Let* $T \cup \{\varphi\}$ *be a finite set of formulae. There is an algorithm that produces a formula* ψ *(given an input* φ *and* T*) such that* ψ *is provable in* $\mathbf{FL_c^+}$ *iff* φ *is provable in* $\mathbf{FL_c}$ *from* T*.*

Acknowledgment

The authors wish to thank Wojciech Buszkowski and anonymous referees for valuable comments and remarks. The work was supported by the grant P202/11/1632 of the Czech Science Foundation and the long-term strategic development financing of the Institute of Computer Science (RVO:67985807).

References

[1] R. V. Book and F. Otto. *String-rewriting Systems*. Springer, 1993.

[2] L. Chottin. Strict deterministic languages and controlled rewriting systems. In H. A. Maurer, editor, *Automata, Languages and Programming*, volume 71 of *Lecture Notes in Computer Science*, pages 104–117. Springer Berlin Heidelberg, 1979.

[3] K. Chvalovský. Undecidability of consequence relation in full non-associative Lambek calculus. *Journal of Symbolic Logic*, to appear.

[4] A. Ciabattoni, N. Galatos, and K. Terui. Algebraic proof theory for substructural logics: Cut-elimination and completions. *Annals of Pure and Applied Logic*, 163(3):266–290, March 2012.

[5] B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 2nd edition, 2002.

[6] R. Freese. Free modular lattices. *Transactions of the AMS*, 261(1):81–91, 1980.

[7] N. Galatos and P. Jipsen. Residuated frames with applications to decidability. *Transactions of the AMS*, 365:1219–1249, 2013.

[8] N. Galatos, P. Jipsen, T. Kowalski, and H. Ono. *Residuated Lattices: An Algebraic Glimpse at Substructural Logics*, volume 151 of *Studies in Logic and the Foundations of Mathematics*. Elsevier, Amsterdam, April 2007.

[9] G. Gentzen. Untersuchungen über das logische Schließen I. *Mathematische Zeitschrift*, 39(1):176–210, 1935.

[10] G. Gentzen. Untersuchungen über das logische Schließen II. *Mathematische Zeitschrift*, 39(1):405–431, 1935.

[11] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, 1st edition, 1979.

[12] R. Horčík. Word problem for knotted residuated lattices. *Journal of Pure and Applied Algebra*, 219(5):1548–1563, May 2015.

[13] P. Jipsen and C. Tsinakis. A survey of residuated lattices. In J. Martínez, editor, *Ordered Algebraic Structures:*, volume 7 of *Developments in Mathematics*. Kluwer, 2002.

[14] M. I. Kanovich. The direct simulation of Minsky machines in linear logic. In J.-Y. Girard, Y. Lafont, and L. Regnier, editors, *Advances in Linear Logic*, number 222 in London Mathematical Society Lecture Note Series. Cambridge University Press, 1995.

[15] O. G. Kharlampovich and M. V. Sapir. Algorithmic problems in varieties. *International Journal of Algebra and Computation*, 5:379–602, 1995.

[16] E. Kiriyama and H. Ono. The contraction rule and decision problems for logics without structural rules. *Studia Logica*, 50(2):299–319, 1991.

[17] Y. Komori. Predicate logics without the structural rules. *Studia Logica*, 45(4):393–404, 1986.

[18] S. A. Kripke. The problem of entailment. *Journal of Symbolic Logic*, 24:324, 1959. abstract.

[19] Y. Lafont. The undecidability of second order linear logic without exponentials. *Journal of Symbolic Logic*, 61(2):541–548, 1996.

[20] P. Lincoln, J. Mitchell, A. Scedrov, and N. Shankar. Decision problems for propositional linear logic. *Annals of Pure and Applied Logic*, 56(1–3):239–311, 1992.

[21] M. Lothaire. *Algebraic Combinatorics on Words*. Number 90 in Encyclopedia of Mathematics and its Applications. Cambridge University Press, April 2002.

[22] B. Surarso and H. Ono. Cut elimination in noncommutative substructural logics. *Reports on Mathematical Logic*, 30:13–29, 1996.

[23] A. Urquhart. The undecidability of entailment and relevant implication. *Journal of Symbolic Logic*, 49(4):1059–1073, 1984.

Institute of Computer Science, Academy of Sciences of the Czech Republic, Pod Vodárenskou věží 2, 182 07 Prague 8, Czech Republic

E-mail address: {chvalovsky,horcik}@cs.cas.cz