Reversible Kleene Lattices*

Paul Brunet

University College London, United Kingdom
paul@brunet-zamansky.fr

Abstract

We investigate the equational theory of reversible Kleene lattices, that is algebras of languages with the regular operations (union, composition and Kleene star), together with the intersection and mirror image. Building on results by Andréka, Mikulás and Németi from 2011, we construct the free representation of this algebra. We then provide an automaton model to compare representations. These automata are adapted from Petri automata, which we introduced with Pous in 2015 to tackle a similar problem for algebras of binary relations. This allows us to show that testing the validity of equations in this algebra is decidable, and in fact ExpSpace-complete.

1998 ACM Subject Classification F.4.3 Formal Languages, F.1.1 Models of Computation, F.3.2 Semantics of Programming Languages.

Keywords and phrases Kleene algebra, Automata, Petri nets, Decidability, Complexity.

Digital Object Identifier 10.4230/LIPIcs.MFCS.2017.66

1 Introduction

We are interested in algebras of languages, equipped with the constants empty language (0), unit language (1, the language containing only the empty word), the binary operations of union (+), intersection ($\cap$), and concatenation ($\cdot$), and the unary operations of Kleene star ($\_^\star$) and mirror image, also called converse, ($\_^\smile$). We call these algebras *reversible Kleene lattices*. Given a finite set of variables X, and two terms e, f built from variables and the above operations, we say that the equation $e = f$ (respectively inequation $e \leq f$) is *valid* if the corresponding equality (resp. containment) holds universally. A *free representation* is a set $\mathcal{M}$ together with a map h from terms to elements of $\mathcal{M}$ such that $e = f$ is valid if and only if h maps e and f to the same element of $\mathcal{M}$.

It is well known that to any term over this syntax, one can associate a regular language, and that comparing regular languages is decidable. In fact, the problem of comparing regular expressions with intersection with respect to regular language equivalence is ExpSpace-complete [12]. The difference with the work presented here is that we are considering equations which are stable under substitution. Formally, this means that we do not interpret the letter a as the singleton language $\{a\}$, but rather as a universally quantified variable ranging over all languages. What is remarkable however is that testing the validity of equations in reversible Kleene lattices is still an ExpSpace-complete problem, as we show in this paper. Several fragments of this algebra have been studied:

Kleene algebra (KA) [9]: if we restricts ourselves to the operators of regular expressions (0, 1, +, $\cdot$, and $\_^\star$), then the free representation is the set of regular languages, with the usual definition of the language of an expression. Testing the validity of equations in KA is thus a PSpace-complete problem [19, 14].

* An extended version of this abstract is available at https://hal.archives-ouvertes.fr/hal-01474911, [5].

Kleene algebra with converse (KAC) [3]: if we add to KA the converse operation, then the free representation consists of regular expressions over a duplicated alphabet, with a letter a' denoting the converse of the letter a. The associated decision problem is still in PSPACE.

Identity-free Kleene lattices (KL$^-$) [1]: this algebra stems from the operators 0, $+$, $\cdot$, $\cap$ and $\_^+$, where the latter is the non-zero iteration. Andréka Mikulás and Németi studied this fragment, and showed that the free representation of this algebra consists of languages of series-parallel graphs, downward closed with respect to some graph preorder. We reformulated their results with Pous [6], and introduced a new class of automata, called Petri automata, able to recognise these languages of graphs. In [6] we provided a decision procedure to compare these automata, thus yielding an EXPSPACE decision procedure for the equational theory of this algebra. It is in fact EXPSPACE-complete, thanks to some simple adaptation of a result by Fürer [12].

The present work is then an extension of identity-free Kleene lattices, by adding unit and mirror image. The addition of mirror image is fairly simple, relying mainly on ideas from [3]. However, the seemingly small addition of 1 yields some complications. In fact, in [1, 6] there is a free representation of Kleene allegories, an algebra over the same signature as reversible Kleene lattices, but whose intended model is binary relations rather than languages. In this context, adding 1 means moving from *series-parallel* graphs to graphs of tree-width 2, which might have cycles. This is a significant problem for automata based decision procedures.

In the context of languages, adding 1 yields other problems. However, the free representation we get for reversible Kleene lattices remains more tractable than that of Kleene allegories. In particular we do not create cycles in series parallel graphs, but rather have to collect additional information. Let us illustrate the kind of reasoning we develop to study these algebras with the following inequation: $c \cdot (1 \cap a) \leq a \cdot c$. On the left hand side (LHS), the term $1 \cap a$ appears. This term is either equal to 1 if the empty word belongs to language a, or 0 otherwise. In the first case, the LHS is equal to c and we have $1 \leq a$, meaning that $c = 1 \cdot c \leq a \cdot c$. In the second case the LHS is equal to 0, which is contained in $a \cdot c$ as well. The key observation here is that the second case does not really matter: in a term build out of concatenations, intersections, converse, variables and units, if 0 appears somewhere then the term will always evaluate to 0 and thus be contained in any other term. The free representations we develop for union-free terms consist of pairs of a representation of a 1-free term and a set of language variables which are assumed to contain the empty word. This allows us to make the reasoning we used above automatic.

Following an approach similar to [6], we construct in Section 2 the free representation of reversible Kleene lattices, and introducing a new Petri net-based automata model we show in Section 3 that testing the validity of equations is a decidable problem, and in fact an EXPSPACE-complete one. We conclude and list some perspectives in Section 4.

Basic definitions and notations

For a pair $p = \langle x, y \rangle$, we denote by $\pi_1(p) = x$ the first projection, and by $\pi_2(p) = y$ the second projection. The set of functions from a set A to a set B is written $A \to B$, and the set of partial functions from A to B is written $A \rightharpoonup B$. The number of elements of a finite set A is written $|A|$. The empty word is denoted by ε, and the set of words over the alphabet Σ is $\Sigma^\star$. If $w = x_1 \dots x_n$ is a word of length n, $w[i, j]$ is the word $x_i \dots x_j$ if $i \leqslant j$, and undefined otherwise. If f is a function from some set X to $\{0, \dots, n\}$, x, y are elements of X, and if $f(x) \leqslant f(y)$, we use the notation $w^f[x, y]$ for the word $w[f(x), f(y)]$.

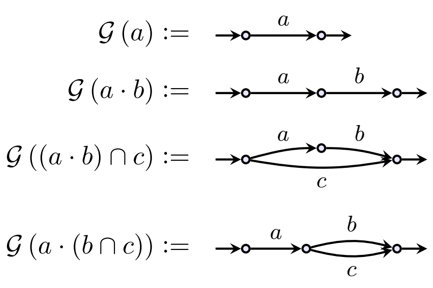

$\mathcal{G}(a) :=$
a
$\mathcal{G}(a \cdot b) :=$
a
b
$\mathcal{G}((a \cdot b) \cap c) :=$
a
b
c
$\mathcal{G}(a \cdot (b \cap c)) :=$
a
b
c

Figure 1 Graphs associated to terms.

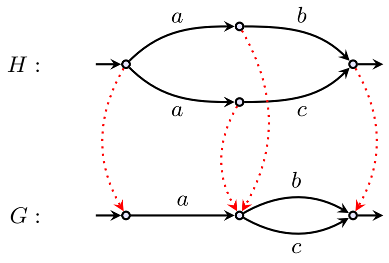


Figure 2 Graph homomorphism.

Let X be a finite set of variables, we define $\dot{A} := A \cup \{a^\smallsmile \mid a \in A\}$ for every subset $A \subseteq X$. The set $\dot{X}$ is called the duplicated alphabet, we let α, β range over $\dot{X}$. Expressions over X are given by the following grammar:

$$e, f ::= 0 \mid 1 \mid x \mid e^\smallsmile \mid e + f \mid e \cdot f \mid e \cap f \mid e^\star. \qquad (x \in X)$$

The set of expressions over X is written $\mathcal{E}\langle X \rangle$. The size of an expression e, written $|e|$, is its number of symbols, *i.e.* the number of vertices in its syntax tree. Most of the time, we will implicitly assume that the converse operator only appears as $x^\smallsmile$, with $x \in X$. This is not restrictive, as every expression can be transformed linearly such that this property holds. Given a second alphabet Σ, an interpretation is a map $\sigma : X \to \mathcal{P}(\Sigma^\star)$ which associates to every variable a a language $\sigma(a)$. This map can be uniquely extended to a homomorphism $\widehat{\sigma} : \mathcal{E}\langle X \rangle \to \mathcal{P}(\Sigma^\star)$ defined inductively:

$$\widehat{\sigma}(0) = \emptyset \qquad \widehat{\sigma}(1) = \{\varepsilon\} \qquad \widehat{\sigma}(a) = \sigma(a) \qquad \widehat{\sigma}(e^\smallsmile) = \widehat{\sigma}(e)^\smallsmile = \{x_n \dots x_1 \mid x_1 \dots x_n \in \widehat{\sigma}(e)\}$$

$$\widehat{\sigma}(e + f) = \widehat{\sigma}(e) \cup \widehat{\sigma}(f) \qquad \widehat{\sigma}(e \cdot f) = \widehat{\sigma}(e) \cdot \widehat{\sigma}(f) \qquad \widehat{\sigma}(e \cap f) = \widehat{\sigma}(e) \cap \widehat{\sigma}(f)$$

$$\widehat{\sigma}(e^\star) = \widehat{\sigma}(e)^\star = \{w_1 \dots w_n \mid w_i \in \widehat{\sigma}(e)\}.$$

We say that $e = f$ (respectively $e \leq f$) is valid, and write $\mathrm{Lang} \models e = f$ (resp. $\mathrm{Lang} \models e \leq f$), when for every interpretation σ, we have $\widehat{\sigma}(e) = \widehat{\sigma}(f)$ (resp. $\widehat{\sigma}(e) \subseteq \widehat{\sigma}(f)$).

It is interesting to note that as decision problems, the validity of equations and that of inequations are equivalent. Indeed, the following equivalences hold:

$$\mathrm{Lang} \models e = f \Leftrightarrow \mathrm{Lang} \models e \leq f \wedge \mathrm{Lang} \models f \leq e \qquad \mathrm{Lang} \models e \leq f \Leftrightarrow \mathrm{Lang} \models e + f = f.$$

2 The free representation of reversible Kleene lattices

2.1 Intuitions

First introduced in the context of relation algebra [2, 11], directed labelled 2-pointed graphs can be used to describe the algebra of languages over the signature $\langle \cdot, \cap \rangle$. First, we associate to every term u over this signature such a graph $\mathcal{G}(u)$. See Figure 1 for examples. The set of such graphs is equipped with a preorder: G is smaller than H if there is a graph homomorphism from H to G. Such a homomorphism is illustrated in Figure 2.

Already, this gives us a clue as to the (in)equational theory: the inequation $u \leq v$ is valid if and only if $\mathcal{G}(u)$ is smaller than $\mathcal{G}(v)$. Moving to identity-free Kleene lattices, *i.e.* to

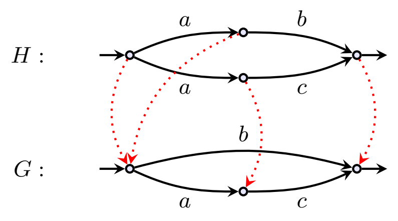


Figure 3 Weak graph morphism: $\langle G, \{a\}\rangle \blacktriangleleft \langle H, \emptyset\rangle$

the signature $\langle 0, +, \cdot, \cap, \_^{+}\rangle$, we associate to every term e a set $\mathcal{G}(e)$ of such graphs, and then take its downward closure $\mathcal{G}(e)\downarrow$ with respect to the preorder. This yields the free representation of this algebra: the equation $e = f$ is valid if and only if $\mathcal{G}(e)\downarrow = \mathcal{G}(f)\downarrow$.

To move from identity free Kleene lattices to reversible Kleene lattices, two steps are necessary: we need to add the converse, and to add the constant 1. The first step is somewhat straightforward, thanks to results by Ésik et al. [3]: they showed that the free representation of the algebra of languages with the regular operations together with converse is simply the set of regular languages over a duplicated alphabet, where we add for every letter a new letter representing its converse. This approach works well in our setting, by considering graphs labelled with the duplicated alphabet.

For the second step, we draw our inspiration from Lemma 3.4 in [1], that established that every term in $\mathcal{E}\langle X\rangle$ is equivalent to a finite sum of terms of the form $(1 \cap a \cap b \ldots) \cdot e$, where $a, b, \dots \in X$ are letters, and 1 does not appear in e. For every interpretation $\sigma : X \to \mathcal{P}(\Sigma^\star)$, if there is some variable $x \in \{a, b, \dots\}$ such that $\varepsilon \notin \sigma(x)$, then the interpretation of $1 \cap a \cap b \cap \dots$ is $\emptyset$. Otherwise, if the empty word is in the interpretation of each of the $a, b, \dots$, then the interpretation is $\{\varepsilon\}$. If we now look at the interpretation of the whole term, this means that:

$$\widehat{\sigma}((1 \cap a \cap b \cap \dots) \cdot e) = \begin{cases} \widehat{\sigma}(e) & \text{if } \forall x \in \{a, b, \dots\}, \varepsilon \in \sigma(x); \\ \emptyset & \text{otherwise.} \end{cases}$$

Consider now an inequation $f_1 \leq f_2$, where $f_i = (1 \cap a_{i,1} \cap \dots \cap a_{i,n_i}) \cdot e_i$ for $i \in \{1, 2\}$. If there exists a variable $x \in \{a_{2,1}, \dots, a_{2,n_2}\} \setminus \{a_{1,1}, \dots, a_{1,n_1}\}$, we can build an interpretation σ such that (1) $\varepsilon \notin \sigma(x)$ and (2) $\widehat{\sigma}(f_1) \neq \emptyset$. The first condition ensures that the image of f_2 will be $\emptyset$, hence the inequation is not valid. Thus for the inequation to be valid, we need that $\{a_{2,1}, \dots, a_{2,n_2}\} \subseteq \{a_{1,1}, \dots, a_{1,n_1}\}$. Furthermore, for every σ such that there is an $a_{1,i}$ whose interpretation does not contain the empty word, the image of f_1 will be $\emptyset$, which is trivially contained in the image of the f_2. We reach the following equivalence: $f_1 \leq f_2$ is valid if and only if (1) $\{a_{2,1}, \dots, a_{2,n_2}\} \subseteq \{a_{1,1}, \dots, a_{1,n_1}\}$ and (2) for every interpretation σ such that the empty word is in the interpretation of every $a_{1,j}$, we have $\widehat{\sigma}(e_1) \subseteq \widehat{\sigma}(e_2)$. This means that we need to compare 1-free expressions under the assumption that certain variables contain the empty word.

This is the intuitions behind what we call weak graphs. Weak graphs are pairs of a graph and a set of test variables. They are equipped with a preorder relation $\blacktriangleleft$, which relates $\langle G, A\rangle$ and $\langle H, B\rangle$ if $B \subseteq A$ and there is a map φ from H to G such that every edge labelled outside of A is preserved, but edges labelled with tests in A are either preserved or contracted. Such a map is shown in Figure 3.

We then have theorems similar to those for identity free Kleene lattices and series parallel graphs, in the sense that for every pair of terms u, v over the syntax $\langle \cdot, \cap, 1, \_^{\smile}\rangle$, if we denote by $\mathcal{WG}(u), \mathcal{WG}(v)$ their associated weak graphs, $u \leq v$ is valid if and only if

$\mathcal{WG}(u) \blacktriangleleft \mathcal{WG}(v)$. Furthermore, if we associate to every expression e in $\mathcal{E}\langle X\rangle$ a downwards closed set of weak graphs ${}^{\blacktriangleleft}[\![e]\!]$, the equation $e = f$ is valid if and only if ${}^{\blacktriangleleft}[\![e]\!] = {}^{\blacktriangleleft}[\![f]\!]$.

2.2 Weak terms

We define the following two sets of terms over the alphabet X:
Ground terms: $u, v \in GT\langle X\rangle ::= 1 \mid a \mid a^{\smile} \mid u \cdot v \mid u \cap v$.
Simple ground terms: $u, v \in GT^{-}\langle X\rangle ::= a \mid a^{\smile} \mid u \cdot v \mid u \cap v$.

We call the variables of the term u, and write $\mathit{var}(u)$, the set of variables $a \in X$ such that a or $a^{\smile}$ appears in u. We call *weak terms* the elements of the set $(GT^{-}\langle X\rangle \cup \{1\}) \times \mathcal{P}(X)$, that is simple ground terms or 1 indexed with a set of test variables. The set of weak terms is written $WT\langle X\rangle$. This set is equipped with two products, denoted by $\bullet$ and $\|$, defined by:

$$1_A \bullet 1_B := 1_{A\cup B} \qquad 1_A \bullet u_B = u_A \bullet 1_B := u_{A\cup B} \qquad u_A \bullet v_B := (u \cdot v)_{A\cup B}$$

$$1_A \parallel 1_B := 1_{A\cup B} \qquad 1_A \parallel u_B = u_A \parallel 1_B := 1_{A\cup B\cup \mathit{var}(u)} \qquad u_A \parallel v_B := (u \cap v)_{A\cup B}.$$

Given an interpretation $\sigma : X \to \mathcal{P}(\Sigma^\star)$, the interpretation $\tilde{\sigma}(u_A)$ of the weak term u_A is either $\emptyset$ if $\exists a \in A : \varepsilon \notin \sigma(a)$, or $\widehat{\sigma}(u)$ otherwise. We define a translation τ from ground terms to weak terms:

$$\tau(u \cdot v) := \tau(u) \bullet \tau(v) \qquad \tau(u \cap v) := \tau(u) \parallel \tau(v) \qquad \forall u \in \{1\} \cup \dot{X},\ \tau(u) := u_\emptyset.$$

This translation is faithful, in the sense that the following holds:

▶ **Lemma 1.** $\forall u \in GT\langle X\rangle, \forall \sigma : X \to \mathcal{P}(\Sigma^\star),\ \widehat{\sigma}(u) = \tilde{\sigma} \circ \tau(u)$.

Proof (Sketch). The proof relies on the fact that for every pair of weak terms x, y we have:

$$\tilde{\sigma}(x \bullet y) = \tilde{\sigma}(x) \cdot \tilde{\sigma}(y) \qquad \tilde{\sigma}(x \parallel y) = \tilde{\sigma}(x) \cap \tilde{\sigma}(y).$$

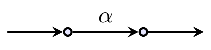
We then conclude by a simple induction on u. For concision, the full proof is omitted here. ◀

We can also define a converse translation $\kappa : WT\langle X\rangle \to GT\langle X\rangle$ which associate to a weak term $u_{\{a_1,\dots,a_n\}}$ the ground term $(1 \cap a_1 \cap \cdots \cap a_n) \cdot u$. It is immediate to check that for every term $x \in WT\langle X\rangle$ and every interpretation σ we have $\tilde{\sigma}(x) = \widehat{\sigma} \circ \kappa(x)$.

2.3 Weak graphs

A *graph* G in our setting is a tuple $\langle V_G, E_G, i_G, o_G\rangle$, where V_G is a finite set of vertices, $E_G \subseteq V_G \times \dot{X} \times V_G$ is a set of labelled and directed edges, and $i_G, o_G \in V_G$ are two vertices, called the input and output of the graph. *Term graphs* must further be series parallel[23], i_G must be the unique source vertex (*i.e.* with no incoming edge), and o_G the unique sink vertex (*i.e.* with no outgoing edge). We let G, H range over graphs. Term graphs can be sequentially composed, by identifying the output of the first graph with the input of the second one, or composed in parallel, by identifying the inputs of both graphs and identifying theirs outputs. These two compositions are respectively denoted by ; and |. The set $\ell(G)$ of labels of a graph G is defined as the set of letters $a \in X$ such that there is an edge in E_G labelled with either a or $a^{\smile}$.

The graph of a simple ground term u, written $\mathcal{G}(u)$, is a term graph defined inductively:

$\mathcal{G}(\alpha) :=$  $\qquad \mathcal{G}(u \cdot v) := \mathcal{G}(u) ; \mathcal{G}(v) \qquad \mathcal{G}(u \cap v) := \mathcal{G}(u) \mid \mathcal{G}(v)$.

We define the graph $\nVdash$ as $\longrightarrow\circ\longrightarrow$. Notice that it is not a term graph, as it is not series parallel. We call *weak graph* a pair whose left part is either a term graph or $\nVdash$, and whose right part is a set of test variables. We denote the weak graph $\langle G, A\rangle$ by G_A. For every weak term x we associate a weak graph $\mathcal{WG}(x)$ as one would expect:

$$\mathcal{WG}(1_A) := \nVdash_A \qquad\qquad \mathcal{WG}(u_A) := \mathcal{G}(u)_A\,.$$

The weak graph G_A is smaller than H_B, written $G_A \blacktriangleleft H_B$, if $B \subseteq A$ and there exists a function $\varphi : V_H \to V_G$ such that $\varphi(i_H) = i_G$, $\varphi(o_H) = o_G$, and for every edge $\langle x, \alpha, y\rangle$ in E_H, either $\langle \varphi(x), \alpha, \varphi(y)\rangle \in E_G$ or $\alpha \in \dot{A}$ and $\varphi(x) = \varphi(y)$. The relation $\blacktriangleleft$ is a preorder. We will show in the next section that for any two ground terms u and v, the following holds:

$$\text{Lang} \models u \leq v \Leftrightarrow \mathcal{WG}(\tau(u)) \blacktriangleleft \mathcal{WG}(\tau(v))\,.$$

The first important lemma is the following. It generalises [1, Lemma 2.5] by including 1 and $\_^{\smile}$, thus moving from series parallel graphs to weak graphs.

▶ **Lemma 2.** *$\forall u \in WT\langle X\rangle$, there exists a word w_u and an interpretation σ_u such that for every $v \in WT\langle X\rangle$, $w_u \in \tilde{\sigma}_u(v) \Leftrightarrow \mathcal{WG}(u) \blacktriangleleft \mathcal{WG}(v)$.*

Proof. Let $\mathcal{WG}(u) = \langle\langle V_u, E_u, i_u, o_u\rangle, A\rangle$. Let $\mu : V_u \to \{1, \ldots, |V_u|\}$ be a bijective map such that $\langle x, \alpha, y\rangle \in E_u \Rightarrow \mu(x) < \mu(y)$[1]. In particular, $\mu(i_u) = 1$ and $\mu(o_u) = |V_u|$. Let $n = 2 \times (|V_u| - 1)$, and Σ_u an alphabet composed of n distinct letters $x_1, \ldots, x_n$.

We define $w_u = x_1 x_2 \ldots x_n$, and $f : V_u \to \{0, \ldots, n\}$ such that $f(x) = 2(\mu(x) - 1)$. Notice that $f(i_u) = 0$ and $f(o_u) = n$. We now define σ_u:

$$\sigma_u(a) := \begin{cases} \left\{w_u^f[x,y] \mid \langle x, a, y\rangle \in E_u\right\} \cup \left\{w_u^f[x,y]^{\smile} \;\middle|\; \langle x, a^{\smile}, y\rangle \in E_u\right\} \cup \{\varepsilon\} & \text{if } a \in A \\ \left\{w_u^f[x,y] \mid \langle x, a, y\rangle \in E_u\right\} \cup \left\{w_u^f[x,y]^{\smile} \;\middle|\; \langle x, a^{\smile}, y\rangle \in E_u\right\} & \text{if } a \notin A \end{cases}$$

Notice that for every $\alpha \in \dot{X}$, $\varepsilon \in \widehat{\sigma}_u(\alpha) \Leftrightarrow \alpha \in \dot{A}$, and that if $x \neq y$ then $w_u^f[x,y] \in \widehat{\sigma}_u(\alpha)$ if and only if $\langle x, \alpha, y\rangle \in E_u$.

Let $v = t_B \in WT\langle X\rangle$. First, suppose that $\exists a \in B \setminus A$. We know that $\varepsilon \notin \sigma_u(a)$, meaning that $\tilde{\sigma}_u(v) = \emptyset$. We also know by definition of $\blacktriangleleft$ that $\mathcal{WG}(u) \not\blacktriangleleft \mathcal{WG}(v)$. Thus the equivalence holds, as both sides are false. In the following, we thus assume that $B \subseteq A$.

If $t = 1$, then $\tilde{\sigma}_u(v) = \{\varepsilon\}$. This means that $w_u \in \tilde{\sigma}_u(v)$ if and only if $w_u = \varepsilon$. By definition, this is equivalent to $n = 0$, which is again equivalent to $|V_u| = 1$ thus to $u = 1_A$. We conclude this case by noticing that the only graph G such that $G_A \blacktriangleleft \nVdash_B$ is $\nVdash$ itself.

The other case is when t is a simple ground term. Then an induction much like in the proof of [1, Lemma 2.5] allows to conclude. We omit this part of the proof here. ◀

The other important lemma is a generalisation of [1, Lemma 2.3]. It will allow us to factor any interpretation of u through the weak graph $\mathcal{WG}(u)$.

▶ **Lemma 3.** *For every simple ground term u, every interpretation $\sigma : X \to \mathcal{P}(\Sigma^\star)$, and every word $w \in \Sigma^\star$ of length n:*

$$w \in \widehat{\sigma}(u) \Leftrightarrow \exists \varphi : V_u \to \{0, \ldots, n\} : \begin{cases} \varphi(i_u) = 0 \wedge \varphi(o_u) = n \\ \langle x, \alpha, y\rangle \in E_u \Rightarrow w^{\varphi}[x,y] \in \widehat{\sigma}(\alpha)\,. \end{cases}$$

It can be proved by a simple induction on u; for concision, we omit this proof.

[1] Remember that both term graphs and $\nVdash$ are directed acyclic graphs.

2.4 Freeness results

We can now establish our first freeness result:

▶ **Theorem 4.** *$\forall x, y \in GT\langle X\rangle$, $\mathcal{WG}(\tau(x)) \blacktriangleleft \mathcal{WG}(\tau(y)) \Leftrightarrow \mathrm{Lang} \models x \leq y$.*

Proof. The statement of the theorem is equivalent to the following, thanks in part to Lemma 1: $\forall x, y \in WT\langle X\rangle, \mathcal{WG}(x) \blacktriangleleft \mathcal{WG}(y) \Leftrightarrow \forall \Sigma, \forall \sigma : X \to \mathcal{P}(\Sigma^\star), \tilde{\sigma}(x) \subseteq \tilde{\sigma}(y)$. We let $x = u_A$ and $y = v_B$, and proceed to prove both implications.

Suppose $\mathcal{WG}(x) \blacktriangleleft \mathcal{WG}(y)$, let σ be an interpretation, and w a word of length n. The case of 1 being trivial, we consider here the case where both u and v are simple ground terms. Assume $w \in \tilde{\sigma}(x)$, then we need to prove that $w \in \tilde{\sigma}(y)$. First notice that because $\tilde{\sigma}(x) \neq \emptyset$ it must be the case that $\forall a \in A, \varepsilon \in \sigma(a)$. By Lemma 3, we have a function $\varphi : V_u \to \{0, \dots, n\}$ such that $\varphi(i_u) = 0$, $\varphi(o_u) = n$, and $\langle x, \alpha, y\rangle \in E_u \Rightarrow w^\varphi[x, y] \in \widehat{\sigma}(\alpha)$. By definition of $\blacktriangleleft$, we also have a function $\psi : V_v \to V_u$ such that $\psi(i_v) = i_u$, $\psi(o_v) = o_u$, and for every edge $\langle x, \alpha, y\rangle$ in E_v, either $\langle \psi(x), \alpha, \psi(y)\rangle \in E_u$ or $\alpha \in \dot{A}$ and $\psi(x) = \psi(y)$. We define $\Phi = \varphi \circ \psi$. Now we may check that $\Phi(i_v) = \varphi(i_u) = 0$; $\Phi(o_v) = \varphi(o_v) = n$; and if $\langle x, \alpha, y\rangle \in E_v$, then either
- $\langle \psi(x), \alpha, \psi(y)\rangle \in E_u$, which means $w^\Phi[x, y] = w^\varphi[\psi(x), \psi(y)] \in \widehat{\sigma}(\alpha)$;
- or $\alpha \in \dot{A}$ and $\psi(x) = \psi(y)$, which entails $w^\Phi[x, y] = \varepsilon \in \widehat{\sigma}(\alpha)$.

Using Lemma 3 again, we get that $w \in \widehat{\sigma}(v)$. Because $B \subseteq A$, we also have that $\tilde{\sigma}(y) = \widehat{\sigma}(v)$. Hence $\tilde{\sigma}(x) \subseteq \tilde{\sigma}(y)$.

For the converse, we now assume that $\mathcal{WG}(x) \not\blacktriangleleft \mathcal{WG}(y)$. Using Lemma 2, we know that $w_x \in \tilde{\sigma}_x(x)$ and that $w_x \notin \tilde{\sigma}_x(y)$. This proves that $\tilde{\sigma}_x(x) \not\subseteq \tilde{\sigma}_x(y)$. ◀

We define the set of weak terms $\llbracket e \rrbracket$ of an expression e by structural induction:

$$\llbracket 0 \rrbracket := \emptyset \qquad \llbracket 1 \rrbracket := \{1_\emptyset\} \qquad \llbracket \alpha \rrbracket := \{\alpha_\emptyset\} \qquad \llbracket e + f \rrbracket := \llbracket e \rrbracket \cup \llbracket f \rrbracket$$

$$\llbracket e \cdot f \rrbracket := \{u \bullet v \mid u \in \llbracket e \rrbracket \wedge v \in \llbracket f \rrbracket\} \qquad \llbracket e \cap f \rrbracket := \{u \parallel v \mid u \in \llbracket e \rrbracket \wedge v \in \llbracket f \rrbracket\}$$

$$\llbracket e^\star \rrbracket := \{u_1 \bullet \dots \bullet u_n \mid n \geqslant 0 \wedge \forall 0 \leqslant i \leqslant n, u_i \in \llbracket e \rrbracket\}$$

The downward closure ${}^{\blacktriangleleft}S$ of a set of weak terms S is the set of weak terms x such that there exists a weak term $y \in S$ satisfying $\mathcal{WG}(x) \blacktriangleleft \mathcal{WG}(y)$. The function ${}^{\blacktriangleleft}$ is a closure operator. The set of downward closed sets of weak terms is the free representation of reversible Kleene lattices:

▶ **Theorem 5.** *$\forall e, f \in \mathcal{E}\langle X\rangle$: ${}^{\blacktriangleleft}\llbracket e \rrbracket \subseteq {}^{\blacktriangleleft}\llbracket f \rrbracket \Leftrightarrow \mathrm{Lang} \models e \leq f$.*

Proof. We use the fact that for every interpretation σ,

$$\widehat{\sigma}(e) = \bigcup_{u \in \llbracket e \rrbracket} \tilde{\sigma}(u) = \bigcup_{u \in {}^{\blacktriangleleft}\llbracket e \rrbracket} \tilde{\sigma}(u).$$

This can be proved using [1, Lemma 2.1], and Lemmas 1 and 2 and Theorem 4.

Suppose ${}^{\blacktriangleleft}\llbracket e \rrbracket \subseteq {}^{\blacktriangleleft}\llbracket f \rrbracket$, and let σ be an interpretation.

$$\widehat{\sigma}(e) = \bigcup_{u \in {}^{\blacktriangleleft}\llbracket e \rrbracket} \tilde{\sigma}(u) \subseteq \bigcup_{u \in {}^{\blacktriangleleft}\llbracket f \rrbracket} \tilde{\sigma}(u) = \widehat{\sigma}(f).$$

For the converse, suppose ${}^{\blacktriangleleft}\llbracket e \rrbracket \not\subseteq {}^{\blacktriangleleft}\llbracket f \rrbracket$. Because ${}^{\blacktriangleleft}$ is a closure operator, this means $\llbracket e \rrbracket \not\subseteq {}^{\blacktriangleleft}\llbracket f \rrbracket$. Let $u \in \llbracket e \rrbracket \setminus {}^{\blacktriangleleft}\llbracket f \rrbracket$. By Lemma 2, we have $w_u \in \tilde{\sigma}_u(u) \subseteq \widehat{\sigma}_u(e)$, but because $u \notin {}^{\blacktriangleleft}\llbracket f \rrbracket$, for every $v \in \llbracket f \rrbracket$, we have $\mathcal{WG}(u) \not\blacktriangleleft \mathcal{WG}(v)$ thus $w_u \notin \tilde{\sigma}_u(v)$. Hence w_u is not in the set $\bigcup_{v \in \llbracket f \rrbracket} \tilde{\sigma}_u(v) = \widehat{\sigma}_u(f)$. ◀

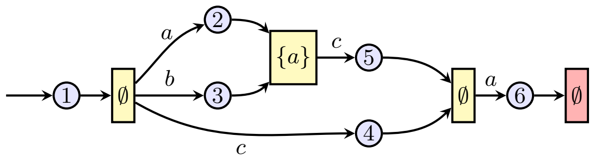


Figure 4 Weak Petri automaton.

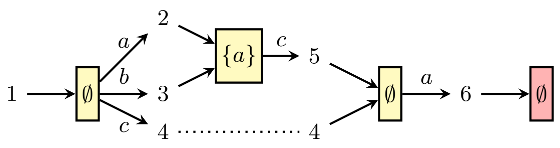
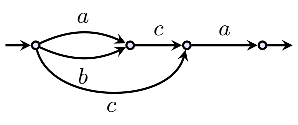


Figure 5 A run R in the automaton of Figure 4.



$$A_R = \{a\}\,.$$

Figure 6 Trace of R.

3 Decidability and complexity

To decide the equational theory of identity-free Kleene lattices, we used Petri automata[6]. This was a new style of automaton, which was designed to recognise sets of series parallel graphs. We modify this model slightly to recognise weak graphs, provide a construction to build automata out of expressions, and an algorithm to decide language containment (up-to closure by $\blacktriangleleft$) for these automata. This algorithm itself is inspired by the simulation algorithm for simple Petri automata. We conclude this section by showing that the problem is complete of the class ExpSpace.

3.1 Weak Petri automata

A weak Petri automaton is a Petri automaton [6, 4] whose transitions are labelled with sets of letters[2]. Formally, an *automaton* $\mathcal{A}$ over the finite alphabet X is a triple $\langle P, T, \iota\rangle$ where P is a finite set of *places*, $\iota \in P$ is the *initial place*, and $T \subseteq \mathcal{P}(P) \times \mathcal{P}(X) \times \mathcal{P}(\dot{X} \times P)$ is a set of *transitions*. Each transition $t \in T$ is composed of three parts: its *input* ${}^{\triangleright}t \subseteq P$, its *set of tests* $\widehat{t} \subseteq X$, and its *output* $t^{\triangleright} \subseteq \dot{X} \times P$. It will also be useful to write $\pi_2(t^{\triangleright})$ for the *set of output places* of t, *i.e.* $\left\{p \in P \;\middle|\; \exists \alpha \in \dot{X} : \langle \alpha, p\rangle \in t^{\triangleright}\right\}$. The transition t is called *final* if $t^{\triangleright} = \emptyset$, and *initial* if ${}^{\triangleright}t = \{\iota\}$.

We will add a few constraints on this definition along the way, but we need more definitions to state them. An example of such an automaton is depicted in Figure 4. The graphical representation used here draws round vertices for places and rectangular vertices for transitions, with the incoming and outgoing arcs to and from the transition corresponding respectively to the inputs and outputs of said transition. The set of tests of a transition is written inside the rectangle. The initial place is denoted by an unmarked incoming arc.

Runs and reachable states

We define the operational semantics of weak Petri automata. Let us fix for the remainder of this section an automaton $\mathcal{A} = \langle P, T, \iota\rangle$. A *state* of this automaton is a set of places. In a given state $S \in \mathcal{P}(P)$, a transition t is *enabled* if ${}^{\triangleright}t \subseteq S$. In this case, we may

[2] In the following, we use the definitions from [4]. They differ slightly from those from [6], despite being overall equivalent.

fire t, leading to a new state $S' = (S \setminus {}^{\triangleright}t) \cup \pi_2\left(t^{\triangleright}\right)$. This will be denoted in the following by $S \xrightarrow{t}_{\mathcal{A}} S'$. We extend this notation to sequences of transitions in the natural way: if $S_0 \xrightarrow{t_1}_{\mathcal{A}} S_1$ and $S_1 \xrightarrow{t_2;\dots;t_n}_{\mathcal{A}} S_n$ then we write $S_0 \xrightarrow{t_1;t_2;\dots;t_n}_{\mathcal{A}} S_n$. In this case we say that $\langle S_0, t_1; t_2; \dots; t_n, S_n\rangle$ is a *valid run*, or simply run, from S_0 to S_n. If $S_0 = \{\iota\}$ then the run is *initial* and if S_n is empty then it is *final*. A run which is both initial and final is called *accepting*. An accepting run of the automaton from Figure 4 is depicted in Figure 5. A state S is reachable in $\mathcal{A}$ if there is an initial run leading to S.

We may now state the first two constraints we impose on automata: if S is reachable in $\mathcal{A}$ and $S \xrightarrow{t}_{\mathcal{A}} S'$, then $(S \setminus {}^{\triangleright}t) \cap \pi_2\left(t^{\triangleright}\right) = \emptyset$, and for each transition $t \in T$, and every triple $\langle p, \alpha, \beta\rangle \in P \times \dot{X} \times \dot{X}$, we have: $\{\langle \alpha, p\rangle, \langle \beta, p\rangle\} \subseteq t^{\triangleright} \Rightarrow \alpha = \beta$. These constraints correspond to the classic Petri net property of safety, also called one-boundedness.

▶ Remark. These constraints are decidable: the set of transitions is finite, and because reachable states are subsets of a fixed finite set, there are only finitely many. Thus checking whether an automaton satisfies these two requirements only entails a finite number of tests.

We introduce some attributes of a run R: its *input* I_R, its *output* O_R, its *excess* E_R, its *tests* A_R, and its *internal labels* Λ_R. Let $R = S_0 \xrightarrow{t_1}_{\mathcal{A}} S_1 \dots \xrightarrow{t_n}_{\mathcal{A}} S_n$ be a valid run in some automaton $\mathcal{A}$. I_R is the set of tokens (places) in S_0 which are consumed during the run; E_R is the rest of the tokens from S_0, those which are not moved; Λ_R is the set of labels appearing in some $t_i^{\triangleright}$ such that the associated token is consumed later on; A_R is the union of the sets of tests of R's transitions; and O_R is the set of outputs which are not consumed in the remainder of the run. Formally:

$$I_R := \{p \in S_0 \mid \exists i : p \in {}^{\triangleright}t_i\} \qquad O_R := \left\{\langle \alpha, p\rangle \;\middle|\; \exists i : \langle \alpha, p\rangle \in t_i^{\triangleright} \wedge \left(\forall j > i, p \notin {}^{\triangleright}t_j\right)\right\}$$

$$A_R := \bigcup_i \widehat{t_i} \qquad E_R := S_0 \setminus I_R \qquad \Lambda_R := \left\{\alpha \;\middle|\; \exists p, \exists i < j : \langle \alpha, p\rangle \in t_i^{\triangleright} \wedge p \in {}^{\triangleright}t_j\right\}.$$

In the example run of Figure 5, we have $I_R = \{1\}$, $E_R = \emptyset$, $O_R = \emptyset$, $A_R = \{a\}$, and $\Lambda_R = \{a, b, c\}$.

Traces

The trace language of an automaton can be obtained by extracting from every accepting run a weak graph, called its *trace*. Consider an accepting run $\langle\{\iota\}, t_0; \dots; t_n, \emptyset\rangle$. The graph of its trace is constructed by creating a vertex k for each transition t_k of the run. We add an edge $\langle k, a, l\rangle$ whenever there is some place q such that $\langle a, q\rangle \in t_k^{\triangleright}$, and t_l is the first transition after t_k in the run with q among its inputs. The set of tests of the trace is A_R. The trace of the run in Figure 5 is presented in Figure 6. The definition we give below is a generalisation for arbitrary valid runs, which coincides with the informal presentation we just gave on accepting runs.

Let $R = \langle S, t_0; \dots; t_n, S'\rangle$ be a run in $\mathcal{A}$. For every k and $p \in \pi_2\left(t_k^{\triangleright}\right) \setminus S'$, we define

$$\nu(k, p) = \min\{l \mid l > k \text{ and } p \in {}^{\triangleright}t_l\}.$$

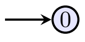
The *trace* of R, denoted by $\mathcal{G}(R)$, is the pair $\langle G_R, A_R\rangle$, where G_R has vertices $V_R = \{0, \dots, n\} \cup S'$ and edges defined by:

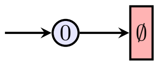
$$E_R = \{\langle k, a, l\rangle \mid \langle a, p\rangle \in t_k^{\triangleright} \text{ and } (l = p \wedge p \in S') \vee (l = \nu(k, p))\}.$$

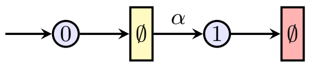
The *language* $\mathcal{L}(\mathcal{A})$ *of an automaton* $\mathcal{A}$ is the set of traces of accepting runs of $\mathcal{A}$. In the following, we will only consider automata such that if $\langle G, A\rangle \in \mathcal{L}(\mathcal{A})$, then either G is either isomorphic to $\nVdash$ or is a term graph: that is, if G_A is a weak graph.

3.2 From expressions to automata

In this section, we show how to build inductively from an expression e an automaton $\mathcal{A}_e$ whose language is $\mathcal{L}(\mathcal{A}_e) = \mathcal{WG}(\llbracket e \rrbracket)$, following [4]. For $0, 1$ and atoms, we give a graphical description of the automata:

$\mathcal{A}_0 :=$ 

$\mathcal{A}_1 :=$ 

$\mathcal{A}_\alpha :=$ 

For the inductive cases, let $\mathcal{A}_e = \langle P_e, T_e, \iota_e \rangle$ and $\mathcal{A}_f = \langle P_f, T_f, \iota_f \rangle$, and suppose $P_e \cap P_f = \emptyset$.

Intuitively, the automaton for $e + f$ is the union of $\mathcal{A}_e$ and $\mathcal{A}_f$, where we copy the initial transitions of $\mathcal{A}_f$ so that they start from ι_e instead of ι_f. Formally:

$$\mathcal{A}_{e+f} = \langle P_e \cup P_f, T_e \cup T_f \cup T, \iota_e \rangle, \text{ where } T = \left\{ \left\langle \{\iota_e\}, \widehat{t}, t^{\triangleright} \right\rangle \;\middle|\; \left\langle \{\iota_f\}, \widehat{t}, t^{\triangleright} \right\rangle \in T_f \right\}.$$

For the product, we want an automaton $\mathcal{A}_{e \cdot f}$ such that $\mathcal{L}(\mathcal{A}_{e \cdot f}) = \mathcal{L}(\mathcal{A}_e) \bullet \mathcal{L}(\mathcal{A}_f)$. This property is satisfied by the automaton $\langle P_e \cup P_f, T_e^+ \cup T_f \cup T, \iota_e \rangle$ where T_e^+ is the set of non-final transitions in T_e, and $T = \{\langle {}^{\triangleright}t, A \cup B, t^{\triangleright} \rangle \mid \langle {}^{\triangleright}t, A, \emptyset \rangle \in T_e \wedge \langle \{\iota_f\}, B, t^{\triangleright} \rangle \in T_f\}$.

Instead of defining directly an automaton for $e^\star$, we give an automaton for the non-zero iteration e^+, and then define $\mathcal{A}_{e^\star}$ to be $\mathcal{A}_{1 \cup e^+}$. Using the last two constructs, the automaton $\mathcal{A}_{e^+}$ is easy to define: $\mathcal{A}_{e^+} = \langle P_e, T_e \cup \{\langle {}^{\triangleright}t, A \cup B, t^{\triangleright} \rangle \mid \langle {}^{\triangleright}t, A, \emptyset \rangle \in T_e \wedge \langle \{\iota_e\}, B, t^{\triangleright} \rangle \in T_e\}, \iota_e \rangle$.

Finally, we then define $\mathcal{A}_{e \cap f}$ to be the automaton $\langle P_e \cup P_f \cup \{\iota\}, T_1 \cup T_2 \cup T_3 \cup T_4, \iota \rangle$, where ι is a fresh place, and:

- T_1 is the set of non-initial, non-final transitions of T_e and T_f;
- T_2 is the set of triples $\left\langle \{\iota\}, \widehat{t_1} \cup \widehat{t_2}, t_1^{\triangleright} \cup t_2^{\triangleright} \right\rangle$ such that t_1 (resp. t_2) is initial but not final in T_e (resp. T_f);
- T_3 is the set of triples $\left\langle {}^{\triangleright}t_1 \cup {}^{\triangleright}t_2, \widehat{t_1} \cup \widehat{t_2}, \emptyset \right\rangle$ such that t_1 (resp. t_2) is final but not initial in T_e (resp. T_f);
- T_4 is the set of triples $\langle \{\iota\}, A, \emptyset \rangle$ such that $1_A \in \llbracket e \cap f \rrbracket$.

This definition is effective, as the set of $A \subseteq X$ such that $1_A \in \llbracket e \rrbracket$ can be computed in space $O\left(|e| \times 2^{|X|}\right)$. Using the proofs for Petri automata as a guideline, it is a simple exercise to check that the correctness of the construction, that is $\mathcal{L}(\mathcal{A}_e) = \mathcal{WG}(\llbracket e \rrbracket)$.

3.3 Comparing automata

The algorithm to compare weak Petri automata relies on the notion of simulation. Similarly to many finite transition systems, the language of an automaton $\mathcal{A}$ is included in that of the automaton $\mathcal{B}$ if $\mathcal{B}$ can simulate $\mathcal{A}$.

▶ **Definition 6** (Simulation). Let $\mathcal{A}_1 = \langle P_1, T_1, \iota_1 \rangle$ and $\mathcal{A}_2 = \langle P_2, T_2, \iota_2 \rangle$ be two automata, we say that $\mathcal{A}_2$ can simulate $\mathcal{A}_1$ if there exists a function $\preccurlyeq$ associating to every subset of X a set of triples from $\mathcal{P}(P_1) \times \mathcal{P}(X) \times \mathcal{P}(P_2 \rightharpoonup P_1)$, such that: *(We denote the fact that the triple $\langle S, B, E \rangle$ is contained in the image by $\preccurlyeq$ of the set A by $S \preccurlyeq_A^B E$.)*

(correspondence) if $S \preccurlyeq_A^B E$ and $\eta \in E$ then $range(\eta) \subseteq S$;

(initialisation) $\{\iota_1\} \preccurlyeq_A^\emptyset \{[\iota_2 \mapsto \iota_1]\}$;

(totality) if $\emptyset \preccurlyeq_A^A E$ then $\exists \eta \in E : dom(\eta) = \emptyset$;

(progress) if $S \preccurlyeq_A^B E$ and $S \xrightarrow{t}_{\mathcal{A}_1} S'$, then $S' \preccurlyeq_A^{B \cup \widehat{t}} E'$, where E' is the set of all η' such that there is a map η in E, and a run R in $\mathcal{A}_2$ from $dom(\eta)$ to $dom(\eta')$ s.t.:

$$I_R = \{p \mid \eta(p) \in {}^{\triangleright}t\} \qquad \Lambda_R \cup A_R \subseteq \dot{A} \qquad \forall \langle \alpha, p \rangle \in O_R, \langle \alpha, \eta'(p) \rangle \in t^{\triangleright}$$

$$\forall p \in E_R, \eta(p) = \eta'(p).$$

▶ **Lemma 7.** *$\mathcal{L}(\mathcal{A}_1) \subseteq {}^{\blacktriangleleft}\mathcal{L}(\mathcal{A}_2)$ if and only if $\mathcal{A}_2$ can simulate $\mathcal{A}_1$.*

Proof. We start by showing the right to left direction: suppose that $\mathcal{A}_2$ can simulate $\mathcal{A}_1$, as witnessed by the function $\preccurlyeq$, and consider an accepting run $R = \langle S_0, t_1; \dots; t_n; S_n\rangle$ in $\mathcal{A}_1$:

$$S_0 = \{\iota_1\} \qquad \forall 1 \leqslant i \leqslant n,\ S_{i-1} \xrightarrow{t_i}_{\mathcal{A}_1} S_i \qquad S_n = \emptyset.$$

We write $B_i = \bigcup_{j<i} \widehat{t_j}$. Using the relations $\preccurlyeq^{B_i}_{A_R}$, we can find a sequence of E_i (for $0 \leqslant i \leqslant n$) such that $S_i \preccurlyeq^{B_i}_{A_R} E_i$, $E_0 = \{[\iota_2 \mapsto \iota_1]\}$, and there is some $\eta_n \in E_n$ which has an empty domain. Backtracking from this η_n using the progress condition allows us to find a sequence of maps $(\eta_i)_{0\leqslant i\leqslant n}$, with domains $(T_i)_{0\leqslant i\leqslant n}$ such that there are valid runs R_i in $\mathcal{A}_2$ from T_{i-1} to T_i, and satisfying:

$$T_0 = \{\iota_2\} \qquad T_n = \emptyset \qquad I_{R_i} = \{p \mid \eta_{i-1}(p) \in {}^{\triangleright}t_i\} \qquad \Lambda_{R_i} \cup A_{R_i} \subseteq \dot{A}_R$$

$$\forall \langle \alpha, p\rangle \in O_{R_i},\ \langle \alpha, \eta_i(p)\rangle \in t_i^{\triangleright} \qquad \forall p \in E_{R_i},\ \eta_{i-1}(p) = \eta_i(p).$$

We now build the run R' by concatenating the R_is. We obtain an accepting run in $\mathcal{A}_2$, whose set of tests is $\bigcup_i A_{R_i} \subseteq A_R$. To any transition t'_j in R', the function φ associates the index i of the run R_i from which this transition was extracted. The function φ witnesses $\mathcal{G}(R) \blacktriangleleft \mathcal{G}(R')$.

For the converse direction, we prove an intermediary result. Let $R = \langle S_0, t_1; \dots; t_n, S_n\rangle$ be an accepting run in $\mathcal{A}_1$ and R' be an accepting run from $\mathcal{A}_2$ such that $\mathcal{G}(R) \blacktriangleleft \mathcal{G}(R')$, with φ as the witnessing function. Notice that if the transition t'_i is a cause of t'_{i+1} in R' (*i.e.* they cannot be exchanged without changing the trace), either $\varphi(i) = \varphi(i+1)$, or $t_{\varphi(i)}$ is a cause of $t_{\varphi(i+1)}$ in R, thus $\varphi(i) < \varphi(i+1)$. This means that we may permute transitions in R' without changing the trace, to obtain a run R'' such that $i < j \Rightarrow \varphi(i) \leqslant \varphi(j)$.

Now, the sets of transitions sharing the same value $\varphi(i)$ are contiguous, meaning that R'' can be split as the sequence of sub-runs $R_1; \dots; R_n$, such that φ maps every transitions in R_i to i. (It may be the case that some of these runs are empty.) As $\mathcal{G}(R) \blacktriangleleft \mathcal{G}(R'')$, we know that $A_{R''} \subseteq A_R$, which means that $\forall i, A_{R_i} \subseteq A_R$. Inside the run R_i, we know that the internal edges of the graph of R_i are labelled with letters from A_R, as both their extremities are mapped to i. This means $\Lambda_{R_i} \subseteq \dot{A}_R$.

We know define the η_i. First, we set $\eta_0(\iota_2) = \iota_1$, and $\forall p \in E_{R_i}, \eta_{i-1}(p) = \eta_i(p)$. If on the other hand $\langle \alpha, p\rangle \in O_{R_i}$, let k be the index in R'' corresponding to the output state of R_i, and $j = \nu_{R''}(p, k)$. As j cannot be in R_i, we know $\varphi(j) > \varphi(k)$. Thus, in the graph of R there is an edge $\langle \varphi(k), \alpha, \varphi(j)\rangle$. By definition of the graph of a run, there must be a pair $\langle \alpha, q\rangle \in t^{\triangleright}_{\varphi(k)}$ such that $\nu_R(q, \varphi(k)) = \varphi(j)$. Then this q is a suitable choice for $\eta_i(p)$.

It is then a simple matter of unfolding the definitions to check that:

$$I_{R_i} = \{p \mid \eta_{i-1}(p) \in {}^{\triangleright}t_i\} \qquad \forall \langle \alpha, p\rangle \in O_{R_i},\ \langle \alpha, \eta_i(p)\rangle \in t_i^{\triangleright} \qquad \forall p \in E_{R_i},\ \eta_{i-1}(p) = \eta_i(p).$$

This means that whenever we have R and R' accepting runs from respectively $\mathcal{A}_1$ and $\mathcal{A}_2$ s.t. $\mathcal{G}(R) \blacktriangleleft \mathcal{G}(R')$, we can find a sequence of η_i satisfying all four conditions of a simulation. Thus, if $\mathcal{L}(\mathcal{A}_1) \subseteq {}^{\blacktriangleleft}\mathcal{L}(\mathcal{A}_2)$, for every reachable state S of $\mathcal{A}_1$, we set $\preccurlyeq^B_A$ to relate S to the set of all maps η such that there is an index i, an accepting run R in $\mathcal{A}_1$, and an accepting run R' of $\mathcal{A}_2$ satisfying (1) $A_R = A$, (2) $\bigcup_{j<i} \widehat{t_j} = B$, (3) $S = S_i$, (4) $\mathcal{G}(R) \blacktriangleleft \mathcal{G}(R')$ and (5) the construction we just provided produces $\eta_i = \eta$. ◀

3.4 Complexity

▶ **Corollary 8.** *The theory of reversible Kleene lattices is* ExpSpace*-complete.*

Proof. The equational theory of identity-free Kleene lattices being already ExpSpace-complete [8, Proposition 10.2], we know the problem at hand to be ExpSpace-hard.

Let $e, f \in \mathcal{E}\langle X\rangle$, we ask whether Lang $\models e \leq f$. By Theorem 5, this reduces to testing if ${}^{\blacktriangleleft}[\![e]\!] \subseteq {}^{\blacktriangleleft}[\![f]\!]$, which is equivalent to $[\![e]\!] \subseteq {}^{\blacktriangleleft}[\![f]\!]$ by the properties of the closure operator. Using the construction in Section 3.2, this is amounts to checking if $\mathcal{L}(\mathcal{A}_e) \subseteq {}^{\blacktriangleleft}\mathcal{L}(\mathcal{A}_f)$. This later question can be decided by looking for a simulation function, thanks to Lemma 7.

We now inspect the space complexity of this method. Let n, m, x be respectively the size of e, the size of f and the size of the alphabet. By analysing each step in Section 3.2, we get that the number of places of $\mathcal{A}_e$ is less than $2 \times n$ (similarly for $\mathcal{A}_f$). The number of transitions is harder to work out from the construction, but because $T \subseteq \mathcal{P}(P) \times \mathcal{P}(X) \times \mathcal{P}(\dot{X} \times P)$, we know it is bounded by $2^{2n+x+2x\times 2n}$. Using Savitch's theorem [22], we only need to show that there is a non-deterministic semi-algorithm to refute the existence of a simulation, which uses only exponential space in n, m and x. Here is such a procedure:

1. choose $A \subseteq X$;
2. start with $S = \{\iota_1\}$, $B = \emptyset$ and $E = \{[\iota_2 \mapsto \iota_1]\}$;
3. if $\langle S, B\rangle = \langle\emptyset, A\rangle$ and E does not contain a map η whose domain is empty return `False`;
4. choose $t \in T_1$ such that ${}^{\blacktriangleright}t \subseteq \pi_1(S)$, fire t from S, and update B as $\widehat{t} \cup B$;
5. update E according to the progress condition in Definition 6;
6. go to step 3.

All of these computations can be performed using exponential space. For instance, S, being a pair of a set of places in $\mathcal{A}_e$ and a set of letters, can be stored in space $2n\log(2n) \times x\log(x)$, and E only needs space $(2n+1)^{2m} \times 2m\log(2n+1)$. ◀

4 Conclusion

We showed that the free representation of reversible Kleene lattices consists of downward closed sets of weak terms, or equivalently of downward closed sets of weak graphs. By considering a suitable variation of Petri automata, and producing an algorithm to decide language containment of these automata, we showed that testing the validity of equations in reversible Kleene lattices is an ExpSpace-complete problem.

The results we obtained here could be naturally extended in a number of ways.

- We would like to add to our model some features of programming languages which have been studied independently, among which tests [16], and nominal structures [13, 18, 17, 7].
- Although Kleene algebra is known not to be finitely axiomatisable [20], several authors have proposed semi-axiomatisations [15, 9, 21]. A complete axiomatisation of Kleene algebra with converse, relative to an axiomatisation of KA, is also known [10]. As far as we know, no axiomatisation of reversible Kleene lattices exists. We believe the free representation we defined in Section 2 could help establishing such an axiomatisation.
- Since we provide here an algorithm, it would be interesting to implement it. Such a procedure could fit in very well in a proof assistant such as Coq.

Although the weak Petri automata introduced in this paper were just a means to an end, we are wondering whether this might be an interesting model of computation in itself. We are confident that we could reuse to technology of boxes introduced in [8, 4] to get a Kleene theorem for these automata. Their semantics could also be reformulated with transitions labelled with weights chosen from a finite lattice (instead of sets of letters from the alphabet).

References

1 Hajnal Andréka, Szabolcs Mikulás, and István Németi. The equational theory of Kleene lattices. *TCS*, 412(52):7099–7108, 2011. doi:10.1016/j.tcs.2011.09.024.

2 Hajnal Andréka and Dmitry A. Bredikhin. The equational theory of union-free algebras of relations. *Alg. Univ.*, 33(4):516–532, 1995. doi:10.1007/BF01225472.

3 Stephen L. Bloom, Zoltán Ésik, and Gheorghe Stefanescu. Notes on equational theories of relations. *Alg. Univ.*, 33(1):98–126, 1995. doi:10.1007/BF01190768.

4 Paul Brunet. *Algebras of Relations: From algorithms to formal proofs.* PhD thesis, Université de Lyon, 2016. URL: https://tel.archives-ouvertes.fr/tel-01455083v1.

5 Paul Brunet. Reversible Kleene lattices. extended abstract, 2017. URL: https://hal.archives-ouvertes.fr/hal-01474911.

6 Paul Brunet and Damien Pous. Petri Automata for Kleene Allegories. In *Proc. LICS*, pages 68–79, July 2015. doi:10.1109/LICS.2015.17.

7 Paul Brunet and Damien Pous. A formal exploration of Nominal Kleene Algebra. In *Proc. MFCS*, 2016. doi:10.4230/LIPIcs.MFCS.2016.22.

8 Paul Brunet and Damien Pous. Petri automata. *Logical Methods in Computer Science*, 2017. submitted. arXiv:1702.01804.

9 John H. Conway. *Regular algebra and finite machines.* Chapman and Hall Mathematics Series, 1971.

10 Zoltán Ésik and Laszlo Bernátsky. Equational properties of Kleene algebras of relations with conversion. *TCS*, 137(2):237–251, 1995. doi:10.1016/0304-3975(94)00041-G.

11 Peter J. Freyd and Andre Scedrov. *Categories, Allegories.* NH, 1990.

12 Martin Fürer. The complexity of the inequivalence problem for regular expressions with intersection. In *Proc. ICALP*, pages 234–245, 1980. doi:10.1007/3-540-10003-2_74.

13 Murdoch J. Gabbay and Vincenzo Ciancia. Freshness and name-restriction in sets of traces with names. In *Proc. FoSSaCS*, pages 365–380. Springer, 2011. doi:10.1007/978-3-642-19805-2_25.

14 Stephen C. Kleene. *Representation of Events in Nerve Nets and Finite Automata*. Memorandum. Rand Corporation, 1951.

15 Dexter Kozen. A completeness theorem for Kleene Algebras and the algebra of regular events. In *Proc. LICS*, pages 214–225. IEEE Computer Society, 1991. doi:10.1109/LICS.1991.151646.

16 Dexter Kozen. Kleene algebra with tests. *Transactions on Programming Languages and Systems*, 19(3):427–443, 1997. doi:10.1145/256167.256195.

17 Dexter Kozen, Konstantinos Mamouras, Daniela Petrisan, and Alexandra Silva. Nominal Kleene coalgebra. In *Proc. ICALP*, pages 286–298. Springer, 2015. doi:10.1007/978-3-662-47666-6_23.

18 Dexter Kozen, Konstantinos Mamouras, and Alexandra Silva. Completeness and incompleteness in nominal Kleene algebra. In *Proc. RAMiCS*, pages 51–66, 2015. doi:10.1007/978-3-319-24704-5_4.

19 Albert R. Meyer and Larry J. Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential space. In *Proc. SWAT*, pages 125–129, 1972. doi:10.1109/SWAT.1972.29.

20 Volodimir N. Redko. On defining relations for the algebra of regular events. *Ukrainskii Matematicheskii Zhurnal*, pages 120–126, 1964.

21 Arto Salomaa. Two complete axiom systems for the algebra of regular events. *J. ACM*, 13(1):158–169, 1966. doi:10.1145/321312.321326.

22 Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of computer and system sciences*, 4(2):177–192, 1970. doi:10.1016/S0022-0000(70)80006-X.

23 Jacobo Valdes, Robert E. Tarjan, and Eugene L. Lawler. The recognition of series parallel digraphs. In *Proc. STOC*, STOC '79, pages 1–12. ACM, 1979. doi:10.1145/800135.804393.