Logical Methods in Computer Science
Volume 20, Issue 3, 2024, pp. 27:1–27:27
https://lmcs.episciences.org/

Submitted Apr. 17, 2023
Published Sep. 30, 2024

COMPLETENESS THEOREMS FOR KLEENE ALGEBRA WITH TESTS AND TOP

DAMIEN POUS [a] AND JANA WAGEMAKER [b]

[a] Plume team, CNRS, LIP, ENS de Lyon
e-mail address: Damien.Pous@ens-lyon.fr

[b] iCIS, Department of Computer Science, Radboud University
e-mail address: jana.wagemaker@ru.nl

Abstract. We prove two completeness results for Kleene algebra with tests and a top element, with respect to guarded string languages and binary relations. While the equational theories of those two classes of models coincide over the signature of Kleene algebra, this is no longer the case when we consider an additional constant "top" for the full element. Indeed, the full relation satisfies more laws than the full language, and we show that those additional laws can all be derived from a single additional axiom. We recover that the two equational theories coincide if we slightly generalise the notion of relational model, allowing sub-algebras of relations where top is a greatest element but not necessarily the full relation.

We use models of closed languages and reductions in order to prove our completeness results, which are relative to any axiomatisation of the algebra of regular events.

For one of our constructions, we extend the concept of finite monoid recognisability to guarded-string languages; this device makes it possible to obtain a PSpace algorithm for the equational theory of binary relations.

1. Introduction

The axiomatic treatment of regular expressions and languages was developed extensively by Conway [Con71], after earlier work of Kleene [Kle56]. Conway asked a difficult question: how to axiomatise the equations between regular expressions that hold under their standard interpretation as formal languages? Redko had proved that every purely equational axiomatisation must be infinite [Red64]. Conway proposed such an infinite axiomatisation, which Krob proved to be complete twenty years later—1991 [Kro91]. Conway had also proposed finite quasi-equational axiomatisations, one of which Kozen proved to be complete also in 1991 [Koz91]—this axiomatisation is now commonly called *Kleene algebra*. By an additional

Key words and phrases: Kleene algebra with tests; closed languages; relation algebra; guarded strings.

This paper is an extended version of the paper with a similar title which appeared in Proc. CONCUR'22 [PW22]; we summarise the additions at the end of the introduction.

This work has been supported by the ERC (CoVeCe, grant No 678157), by the LABEX MILYON (ANR-10-LABX-0070), within the program ANR-11-IDEX-0007, and by the project 'Mode(l)s of Verification and Monitorability' (MoVeMent) (grant No 217987) of the Icelandic Research Fund.

DOI:10.46298/LMCS-20(3:27)2024

remark of Boffa [Bof90], this latter completeness result can also be obtained as a consequence of Krob's completeness result. In the end, all finite quasi-equational axiomatisations proposed by Conway, as well as a few other ones, are actually complete [Kro91, Bof95].

In symbols, writing $[\![e]\!]$ for the language of a regular expression e and $\mathsf{KA} \vdash e = f$ when the equation $e = f$ is derivable in any of the aforementioned axiomatisations, we have that for all regular expressions e, f,

$$\mathsf{KA} \vdash e = f \iff [\![e]\!] = [\![f]\!]$$

The above equivalence extends with two more clauses. When an equation is derivable, it must hold in all models of the chosen axiomatisation. These include in particular language models (LANG) and relational models (REL), for which we actually have an equivalence: writing $\mathcal{X} \models e = f$ when the equation $e = f$ holds in all members of a class of models $\mathcal{X}$, we actually have:

$$\mathsf{KA} \vdash e = f \iff \mathsf{REL} \models e = f \iff \mathsf{LANG} \models e = f \iff [\![e]\!] = [\![f]\!]$$

Completeness w.r.t. LANG is immediate given the previous equivalence: the language interpretation of a regular expression lies in LANG. This is less obvious for REL: completeness comes from a nice trick due to Pratt showing that every member of LANG embeds into a member of REL [Pra80, third page].

As an immediate consequence of the above equivalence, the equational theory of REL (or LANG) is decidable—more precisely, PSpace-complete.

The above-mentioned results apply to the *regular* operations and constants: composition, union, Kleene star, identity, emptiness. A natural question is whether they extend to other operations or constants, such as intersection, converse, fullness. The case of converse was dealt with by Ésik et al.: the equational theories of REL and LANG differ in the presence of converse but both can be axiomatised [BÉS95, ÉB95], and they remain PSpace-complete [BP16]. The case of intersection (with or without converse or the various constants) is significantly more difficult, and remains partly open, see [AMN11, BP15, Nak17, DP18]. In this paper we focus on the addition of a constant $\top$, interpreted as the full language in LANG and as the full relation in REL.

The usefulness of adding such a constant was demonstrated recently in the context of Kleene algebras with tests (KAT) [KS96, CKS96, Koz97], to model *incorrectness logic* [ZdAG22]. Indeed, while KAT alone makes it possible to model Hoare triples for partial correctness [Koz00], the addition of a full element makes it possible to compare the (co)domains of relations, and thus to encode *incorrectness triples* [O'H20, Section 5.3]. KAT with a top element was also used earlier, as an intermediate structure to characterise a semantics for abnormal termination [Mam17, Definition 12].

The theory of KAT was developed extensively by Kozen et al., and it is very similar to that of Kleene algebra. For instance, the above equivalences extend to

$$\mathsf{KAT} \vdash e = f \iff \mathsf{REL} \models e = f \iff \mathsf{GSL} \models e = f \iff [e] = [f]$$

There, GSL is the generalisation of language models (LANG) to *guarded string* language models, and $[e]$ denotes the guarded string language interpretation of a regular expression with tests e. Here also, we get decidability in PSpace, using an appropriate generalisation of finite automata on words to finite automata on guarded strings [CKS96, Koz17].

Dealing with Kleene algebra with tests has important applications in program verification: they make it possible to represent and reason about the big-step semantics of while programs,

algebraically. This was used for instance to analyse compiler optimisations [KP00]. The decidability result was also implemented in proof assistants such as Coq and Isabelle/HOL, in order to automate some reasoning steps about binary relations and Hoare logic on while programs [Pou13, KN12].

Like in [ZdAG22] the problem we consider in this paper is that of adding a constant $\top$. As expected, one should consider an axiom expressing that $\top$ is a greatest element:

$$x \leq \top \tag{T}$$

(Where $x \leq y$ is a shorthand for $x + y = y$.) Together with the Kleene algebra axioms, axiom (T) yields a complete axiomatisation w.r.t. language models: we sketched a proof for the case without tests in [PRW21, Example 3.4], which we make fully explicit here in Section 3 (Theorem 3.6). This proof gives us as a byproduct that the equational theory of Kleene algebras (with tests and) with a greatest element remains PSPACE-complete.

Unfortunately, the previous axiom is not enough to deal with relational models. In fact, in the presence of $\top$, the equational theories of LANG (or GSL) and REL differ. Indeed, there are laws such as $\top x \top y \top = \top y \top x \top$ [Pou18, page 13], or $\top x \top x = \top x$ [ZdAG22, page 14], which are valid in REL, but not in LANG.

In the present paper, we show that it suffices to further add the following axiom in order to obtain a complete axiomatisation for REL (Theorem 4.21):

$$x \leq x \cdot \top \cdot x \tag{F}$$

This inequation is mentioned in [ZdAG22, page 14]; it holds in relational models, but not in language ones. Thanks to (T), axiom (F) may be seen as a consequence of Ésik et al.'s axiom $x \leq x \cdot x^\circ \cdot x$ for dealing with converse $(\cdot^\circ)$ in relational models [ÉB95, BÉS95]. How to use axiom (F) in an equational proof is not so intuitive: it does not give rise to a natural notion of normal form, and it must often be used in conjunction with (T) in order to compensate for the fact that it duplicates subterms. For instance here is how we can prove the first of the aforementioned laws:

$$\begin{aligned}
\top x \top y \top &\leq \top x \top y \top \ \top \ \top x \underline{\top y \top} && \text{(by (F))}\\
&\leq \top x \top y \underline{\top \top \top} x \top && \text{(by (T))}\\
&\leq \underline{\top x \top} y \top x \top && \text{(by (T))}\\
&\leq \top y \top x \top && \text{(by (T))}
\end{aligned}$$

(We wrote compositions by juxtaposition, skipped the associativity steps, and underlined the subterms to be simplified by axiom (T)—the converse inequation is derived symmetrically.) Our completeness proof actually goes via a factorisation property (Proposition 4.8) intuitively asserting that one can always proceed in this way to reason about star-free expressions: expand the expressions using (F) a number of times, then remove spurious subterms using (T). Combining such a technique together with Kleene algebra reasoning for star is the second challenge we address in the present work.

To get a grasp on the difficulties, the reader may try to find a proof of the following valid law of REL, using KA and axiom (F):

$$(aaa)^* \leq (aaa)^* \top (aa)^* + (aa)^* a \top (aaa)^* \tag{$\star$}$$

(Note that there is an easy proof using KA and axiom (T), which however breaks with the following variant: $b(aaa)^* b \leq b((aaa)^* b \top b (aa)^* + (aa)^* ab \top b (aaa)^*) b$.) We give a solution to this exercise in Section 4.3.

Finally, we show that the difference between the equational theories of language and relational models can be blurred if we slightly generalise the notion of relational model, allowing $\top$ to be any greatest relation rather than the full one[1] (Corollary 5.2).

We prove our two main theorems using the concept of *closed language model* for Kleene algebra with hypotheses [DKPP19], and the reduction technique made explicit in [PRW21, KBS+20][2]. Intuitively, we establish reductions from KAT with (T) and KAT with (T, F) to plain KAT, so that we can deduce completeness and decidability of the former theories from completeness and decidability of the latter one.

While the first reduction is relatively straightforward—this is a syntactical linear reduction, the second one is not. We exploit the aforementioned factorisation result (Proposition 4.8) and the theory of language recognition by finite monoids [Eil74, Sak09] in order to show that regular languages are preserved by a certain closure operation, and that this preservation property can be justified algebraically (Proposition 4.20). Doing so for Kleene algebra with tests first requires us to extend such ideas to deal with guarded string languages (Section 4.2.2). Moreover, in order to establish the correspondence between the closed languages used there and relational models, we resort to a graph theoretical characterisation of the equational theory of REL (Theorem 4.5, whose main ingredients date back to the works of Freyd and Scedrov [FS90, page 208] and Andréka and Bredikhin [AB95, Theorem 1]).

Differences with the version in Proc. CONCUR'22. The three main differences with [PW22] are the following: 1/ we deal with Kleene algebra with tests rather than plain Kleene algebra; 2/ we use a technique based on finite monoids rather than on finite automata to deal with axiom (F) in Section 4.2; 3/ we provide a PSpace algorithm for the equational theory, a problem which we had left open.

For 1/ we need to move from word languages to guarded string languages. This change is pervasive but the development scales smoothly without the need for major conceptual changes. For instance, when it comes to the graph-theoretical characterisation (Theorem 4.5), it suffices to add labels to graph vertices. In fact, once the definitions are properly extended, the proofs remain almost unchanged.

Contribution 2/ is entirely new. It makes it possible to avoid the slightly cumbersome automata construction we were using in [PW22, Section 4.2] (which would have been slightly painful to extend to guarded string automata). It is this new construction based on monoids which lead us to the PSpace algorithm we present in Section 4.4 (Contribution 3/). While this idea is conceptually simple once we know the tools from monoid-based recognition of regular languages, some care is required here since we deal with guarded string languages: we have to develop the premises of a theory of monoid-based recognition of guarded string languages. This is why we first explain the idea on plain regular languages (Section 4.2.1).

Our results readily apply to Kleene algebra with top but without tests (cf. Remark 2.1 below). Nevertheless the reader not interested in tests and guarded strings may find it easier to read [PW22] except for its section 4.2, and then jump to Sections 4.2.1, 4.3 and 4.4 from the present paper.

[1] Considering subalgebras where only certain relations are kept, since otherwise the only greatest relation is the full one.

[2] Such a technique is somehow implicit in Kozen and Smith's completeness proof for KAT [KS96] and Ésik et al.'s completeness proof for Kleene algebra with converse [BÉS95, ÉB95].

Outline. We setup and recall basic notation for regular expressions, formal languages and universal algebra in Section 2. Then we deal with guarded string language models in Section 3, and relational models in Section 4. While the language case was already sketched in [PRW21, Example 3.4] (without tests), we find it useful to treat it explicitly here, before dealing with the more involved case of relations: it illustrates the reduction method in a simpler setting, and we build on the reduction for languages to establish the reduction for relations. We finally prove completeness with respect to generalised relational models in Section 5.

2. Preliminaries

Given a set X, we write X^* for the set of *words* over X: finite sequences of elements of X. We let u, v range over words, we write ϵ for the empty word, uv for the concatenation of two words u, v, and u^i for the concatenation of i copies of a word u. We let e, f range over *regular expressions over* X, generated by the following grammar:

$$e, f ::= e + f \mid e \cdot f \mid e^* \mid 0 \mid 1 \mid x \in X$$

We sometimes omit the dots in regular expressions, writing, e.g., ab^* for $a \cdot b^*$. A *language* is a set of words. As usual, we associate a language $[\![e]\!]$ to every regular expression e, the *language of* e.

We fix a finite set Σ of *letters*, ranged over using a, b, and a finite set At of *atoms*, ranged over using α, β, γ. We write $\Sigma_\top$ for the set Σ extended with a new element $\top$ called *top*.

We call *expressions* the regular expressions over $\Sigma_\top \uplus \mathsf{At}$. We call *plain regular expressions* the regular expressions over Σ. We shall sometimes see words over $\Sigma_\top \uplus \mathsf{At}$ as expressions. E.g., the word $\alpha a \beta \top \gamma$ can be seen as the expression $\alpha \cdot a \cdot \beta \cdot \top \cdot \gamma$.

We consider signatures $S \triangleq \{+_2, \cdot_2, {}^*{}_1, 0_0, 1_0\} \cup \{\alpha_0 \mid \alpha \in \mathsf{At}\}$ and $S_\top \triangleq S \cup \{\top_0\}$. In those signatures, there is a constant symbol for every atom $\alpha \in \mathsf{At}$.

Expressions form the free $S_\top$-algebra over Σ. Given an $S_\top$-algebra A and a valuation $\sigma \colon \Sigma \to A$, we write $\hat{\sigma}$ for the unique homomorphism extending σ to expressions. Note that $\hat{\sigma}(\top) = \top_A$ and $\hat{\sigma}(\alpha) = \alpha_A$ by definition: top and atoms are constants, not variables.

Given a class $\mathcal{X}$ of $S_\top$-algebras and two expressions e, f, we write $\mathcal{X} \models e = f$ if for all members A of $\mathcal{X}$ and all valuations $\sigma \colon \Sigma \to A$, we have $\hat{\sigma}(e) = \hat{\sigma}(f)$.

An *equation* is a pair of expressions e, f, written $e = f$. We write $e \leq f$, an *inequation*, as a shorthand for the equation $e + f = f$. An *axiomatisation* is a set of equations (or implications between equations). Given such a set $\mathcal{E}$, we write $\mathcal{E} \vdash e = f$ when the equation $e = f$ is derivable from $\mathcal{E}$ using the rules of equational reasoning (where letters from Σ appearing in the equations of $\mathcal{E}$ can be substituted by arbitrary terms).

We let KA stand for any axiomatisation over plain regular expressions which is sound and complete w.r.t. the regular language interpretation, i.e., such that for all plain regular expressions e, f, we have[3]

$$\mathsf{KA} \vdash e = f \iff [\![e]\!] = [\![f]\!]$$

As explained in the introduction, valid candidates for KA include Conway's infinite but purely equational axiomatisation [Con71, page 116] (proved complete by Krob [Kro91]),

[3]Actually, we require slightly more if the axiomatisation contains implications: those implications should be valid in the models of languages and binary relations.

Kozen's Kleene algebras [Koz91], left-handed Kleene algebras [KS12, DDP18], and Boffa's algebras [Bof95].

Also note that the above requirement is equivalent to the following one, since $L \subseteq K$ iff $L \cup K = K$ for all languages L, K:

$$\mathsf{KA} \vdash e \leq f \iff [\![e]\!] \subseteq [\![f]\!]$$

Let KAT, *Kleene algebra with tests*, be the union of KA and the following equations:

$$\sum_{\alpha \in \mathsf{At}} \alpha = 1 \qquad\qquad \alpha \cdot \beta = 0 \quad (\forall \alpha \neq \beta) \tag{A}$$

Note that we can deduce $\mathsf{KAT} \vdash \alpha \cdot \alpha = \alpha$ for all atoms α, by neutrality of 1 and distributivity.

A *guarded string (over an alphabet X)* is a word over $X \uplus \mathsf{At}$ starting with an atom, alternating between atoms and elements in X, and ending with an atom. The *length* of a guarded string is the number of X-elements in it. For instance, α, $\alpha x \beta$, $\alpha x \beta y \gamma$ are guarded strings of respective lengths 0, 1, and 2, when $x, y \in X$. We write $\mathcal{GS}_X$ for the set of guarded strings over X, which is in bijection with $(\mathsf{At} \times X)^* \times \mathsf{At}$.

The *coalesced product* is a partial binary operation on guarded strings: if $u = u'\alpha$ and $v = \beta v'$ are two guarded strings, then their coalesced product $u \diamond v$ is defined if $\alpha = \beta$, in which case $u \diamond v \triangleq u' \alpha v'$.

A *guarded string language* is a set of guarded strings. The *guarded string language of an expression e* is defined as

$$[e] \triangleq \mathsf{gs}[\![e]\!]$$

where gs is the following function from languages over $\Sigma_\top \uplus \mathsf{At}$ to guarded string languages over $\Sigma_\top$:

$$\begin{aligned} \mathsf{gs} \colon \mathcal{P}((\Sigma_\top \uplus \mathsf{At})^*) &\to \mathcal{P}(\mathcal{GS}_{\Sigma_\top}) \\ L &\mapsto \left\{ \alpha_0 a_0 \dots a_{n-1} \alpha_n \mid \exists i_0, \dots, i_n \in \mathbb{N},\ \alpha_0^{i_0} a_0 \dots a_{n-1} \alpha_n^{i_n} \in L \right\} \end{aligned}$$

For instance, if At consists of three distinct atoms α, β, γ, then

$$[(\alpha a \beta + \beta b \gamma)^*] = \{\alpha, \beta, \gamma, \alpha a \beta, \beta b \gamma, \alpha a \beta b \gamma\}$$

In the absence of top, KAT is sound and complete w.r.t. the guarded string language interpretation: for all expressions without top, we have

$$\mathsf{KAT} \vdash e = f \iff [e] = [f] \tag{†}$$

$$\mathsf{KAT} \vdash e \leq f \iff [e] \subseteq [f] \tag{‡}$$

(This is essentially [KS96, Theorem 8], even though we work with an abstract version of KAT here, cf Remark 2.2 below.) For expressions without top, it is also known that KAT is sound and complete with respect to both relational and guarded string language models [KS96, Theorem 7], which we define in the following sections. The point of this work is to deal with the constant top.

Remark 2.1. By choosing a singleton set $\{*\}$ for At, we recover the case of plain Kleene algebra with top we covered in [PW22]. Indeed, in that case, the first axioms in (A) reads as $* = 1$ so that atoms become redundant in the syntax of expressions, guarded strings over X are in one-to-one correspondence with words over X, and accordingly, guarded string languages are just standard word languages.

Remark 2.2. In the literature, KAT is usually presented as a two-sorted system: one sort for *programs* forming a Kleene algebra, and one sort for *tests* forming a Boolean algebra which embeds as a lattice into the former sort. Given a finite set T of test variables, every element of the free Boolean algebra over T can be represented as a disjunction of atoms in $\mathsf{At} \triangleq 2^T$. This idea makes it possible to normalise standard KAT expressions in such a way that all tests are atoms, giving rise to the syntax and axioms we use in the present paper. This idea is already there in the completeness proof of KAT [KS96]; axioms (A) appear explicitly in works about NetKAT [AFG+14, Figure 6]; we gave a formal reduction between the two presentations in [PRW21, Section 4.2]. We prefer this setup because it is single-sorted and slightly more abstract (e.g., we could imagine models where the set of atoms is not of the form 2^T.) Note that when there are no tests in standard KAT (i.e., $T = \emptyset$), we fall back into the special case discussed in Remark 2.1: $2^\emptyset$ is a singleton.

3. Guarded string language models

Let X be an alphabet and let L, K range over guarded string languages on the alphabet X. These form an $S_\top$-algebra with the operations defined as follows:

$$L + K \triangleq L \cup K \qquad 0 \triangleq \emptyset \qquad \top \triangleq \mathcal{GS}_X$$
$$L \cdot K \triangleq \{u \diamond v \mid u \in L \ \wedge \ v \in K\} \qquad 1 \triangleq \{\alpha \mid \alpha \in \mathsf{At}\} \qquad \alpha \triangleq \{\alpha\}$$
$$L^* \triangleq \bigcup_{n \in \mathbb{N}} L^n \qquad L^0 \triangleq 1 \qquad L^{i+1} \triangleq L \cdot L^i$$

(That is, $+$ is set-theoretic union, 0 and $\top$ are the empty and full languages, respectively, $\cdot$ is guarded string language concatenation, via coalesced product, 1 contains all guarded strings consisting of a single atom, and $\cdot^*$ is obtained via iteration.) We write GSL for the class of all $S_\top$-algebras of the above shape.

Fact 3.1. The guarded string language interpretation of expressions, $[\cdot] = \mathsf{gs}[\![\cdot]\!]$, is the unique S-algebra homomorphism satisfying $[a] = \{\alpha a \beta \mid \alpha, \beta \in \mathsf{At}\}$ for all $a \in \Sigma_\top$. This is not an $S_\top$-algebra homomorphism, since $[\top] = \{\alpha \top \beta \mid \alpha, \beta \in \mathsf{At}\} \subsetneq \mathcal{GS}_{\Sigma_\top} = \top$.

Let KAT_T, *KAT with a top element*, denote the union of the axioms from KAT and axiom (T). We prove in this section that KAT_T is sound and complete for GSL. Following the strategy from [DKPP19, PRW21], the first step consists of defining the closure operation below, according to the axiom (T) we add to KAT:

Definition 3.2 (Language closure C_T)**.** Given two guarded strings u, v over $\Sigma_\top$, we write $u \leftsquigarrow_T v$ if $u = l \diamond w \diamond r$ and $v = l \top r$ for some guarded strings l, w, r. Given a guarded string language L over $\Sigma_\top$, we call *T-closure of L* the following guarded string language

$$C_T(L) \triangleq \{u \mid u \leftsquigarrow_T^* v \text{ for some } v \in L\}$$

C_T is indeed a closure operator, and $C_T(L)$ may alternatively be described as the set of guarded strings obtained by replacing occurrences of $\top$ in a guarded string of L with arbitrary guarded strings (over $\Sigma_\top$).

For instance, we have $\alpha a \alpha \top \beta \rightsquigarrow_T \alpha a \alpha b \gamma c \beta$ (by choosing $w = \alpha b \gamma c \beta$), and $\alpha \top \alpha \rightsquigarrow_T \alpha$ (by choosing $w = \alpha$). Also observe that in a guarded string with shape $u \alpha \top \beta v$, we cannot replace $\top$ with a guarded string of length zero unless $\alpha = \beta$.

Lemma 3.3. *C_T is an $S_\top$-algebra homomorphism.*

Proof. The only interesting case is that of composition, which amounts to showing that for all guarded strings u, v, w,

$$u \leftsquigarrow_T v \diamond w \quad \text{iff} \quad \text{there are } v', w' \text{ s.t. } u = v' \diamond w' \text{ and } \begin{cases} \text{either } v' \leftsquigarrow_T v \text{ and } w' = w, \\ \text{or } v' = v \text{ and } w' \leftsquigarrow_T w \end{cases}$$

This follows from the fact that our rewriting relation $\leftsquigarrow_T$ replaces single letters. □

Definition 3.4 (Expression closure r)**.** Let r be the unique S-algebra homomorphism on expressions such that $r(a) = a$ for all letters $a \in \Sigma$, and $r(\top) = \mathbf{\Sigma}^*_\top$ (where $\mathbf{\Sigma}^*_\top$ is an expression for the full guarded string language $\mathcal{GS}_{\Sigma_\top}$—e.g., $(a + b + \cdots + \top)^*$).

Proposition 3.5. *For all expressions e, we have*

(i) $[r(e)] = C_T[e]$*, and*
(ii) $\mathsf{KAT}_T \vdash e = r(e)$*.*

Proof.

(i) $[r(\cdot)]$ and $C_T[\cdot]$ are S-algebra homomorphisms, and they agree on $\Sigma_\top$.
(ii) We proceed by induction on e; the only interesting case is when $e = \top$, for which we have $\mathsf{KAT}_T \vdash r(\top) \leq \top$ by axiom (T), and $\mathsf{KAT}_T \vdash \top \leq r(\top)$ by completeness of KAT (‡), since $[\top] \subseteq \mathcal{GS}_{\Sigma_\top} = [r(\top)]$. □

Theorem 3.6. *For all expressions e, f, we have*

$$\mathsf{GSL} \models e = f \iff C_T[e] = C_T[f] \iff \mathsf{KAT}_T \vdash e = f$$

Proof. We have

$$\begin{aligned}
& \mathsf{GSL} \models e = f \\
\Rightarrow\ & C_T[e] = C_T[f] && (C_T[\cdot] \text{ is an interpetation into a member of } \mathsf{GSL}, \text{ by Lemma 3.3}) \\
\Leftrightarrow\ & [r(e)] = [r(f)] && (\text{Proposition 3.5(i)}) \\
\Leftrightarrow\ & \mathsf{KAT} \vdash r(e) = r(f) && (\text{completeness of KAT } (\dagger)) \\
\Rightarrow\ & \mathsf{KAT}_T \vdash e = f && (\text{transitivity and Proposition 3.5(ii)}) \\
\Rightarrow\ & \mathsf{GSL} \models e = f && (\text{soundness of } \mathsf{KAT}_T \text{ axioms w.r.t. } \mathsf{GSL})
\end{aligned}$$

(In the last step, soundness w.r.t. GSL comes from our assumption about KA, and a trivial verification for axioms (A) and (T).) □

The first equivalence in the statement of the above theorem could be obtained in a more direct way, without resorting to completeness of some axiomatisation. Moreover the right-to-left implication of the second equivalence is an instance of a general property of closed language models [DKPP19, Theorem 2]—duly generalised to the guarded string case. The reduction r is used only for the left-to-right implication of this second equivalence.

According to the above proof, we could complete the statement with "$\ldots \iff [r(e)] = [r(f)]$". Doing so gives us a PSpace algorithm: compute the expressions $r(e)$ and $r(f)$, and compare them for guarded string language equivalence [CKS96].

Remark 3.7. Note that it is crucial that $r(\top)$ be defined as an expression $\mathbf{\Sigma}^*_\top$ for the full guarded string language over $\Sigma_\top$ rather than over Σ alone. Otherwise, we would equate $\mathbf{\Sigma}^*$ and $\top$, while those are different in GSL (e.g., for a counterexample when $\Sigma = \{a, b\}$, interpret both a and b as the empty language on some non-empty alphabet).

4. Relational models

Given a set X, a *relation on* X is a set of pairs of elements from X. We let R, S range over such relations, whose set is written $\mathcal{P}(X \times X)$, and we write $x\, R\, y$ for $\langle x, y\rangle \in R$. Given a function $p\colon X \to \mathsf{At}$, relations on X form an $S_\top$-algebra with the operations defined as follows:

$$\begin{aligned} R + S &\triangleq R \cup S \\ R \cdot S &\triangleq \{\langle x, z\rangle \mid \exists y \in X,\ x\, R\, y \ \wedge\ y\, S\, z\} \\ R^* &\triangleq \{\langle x_0, x_n\rangle \mid \exists n \in \mathbb{N}, x_1, \ldots, x_{n-1},\ \forall i < n,\ x_i\, R\, x_{i+1}\} \\ 0 &\triangleq \emptyset \\ 1 &\triangleq \{\langle x, x\rangle \mid x \in X\} \\ \top &\triangleq X \times X \\ \alpha &\triangleq \{\langle x, x\rangle \mid p(x) = \alpha\} \end{aligned}$$

In words, $+$ is set-theoretic union, $\cdot$ is relational composition, $\cdot^*$ is reflexive transitive closure, 0, 1 and $\top$ are the empty, identity and full relations, respectively. The function p is only used to define the constants α. The idea is that p describes a partition of X, using atoms to name the equivalence classes. The relation α consists of the sub-identity relation selecting precisely the elements whose equivalence class is named α.

We write REL for the class of all $S_\top$-algebras of the above shape.

Remark 4.1. Note that this definition covers the standard way of interpreting Kleene algebra with tests expressions into relations on a set X. Recall Remark 2.2. If we start from a set T of test variables, then an interpretation $v\colon T \to \mathcal{P}(X)$ of test variables into predicates is the same as a function $p\colon X \to \mathsf{At}$ as above when setting $\mathsf{At} \triangleq 2^T$. Furthermore, the standard interpretation under v of an atom α (seen as conjunction of literals in T) coincides with the one given in the above definition.

In particular, while we set atoms as constants in our signatures, the ability to let p vary in members of REL gives them back their original status of variables.

Let KAT_F, *KAT with a full element*, denote the union of the axioms from KAT_T and axiom (F). Let us emphasise that despite the abbreviation, KAT_F extends KAT_T and thus contains axiom (T). We prove in this section that KAT_F is sound and complete for REL. The proof consists of two parts. First we characterise the equational theory of REL in terms of closed guarded string languages (Section 4.1, Proposition 4.9), then we use reductions to show completeness of KAT_F w.r.t. this closed language interpretation and obtain our main result (Section 4.2, Theorem 4.21).

4.1. Characterisation via closed guarded string languages.

From now on, all guarded strings and associated languages are over $\Sigma_\top$. We start by extending the previous closure function (Definition 3.2), in order to take into account the new axiom (F):

Definition 4.2 (Language closure C_F)**.** Given two guarded strings u, v, we write $u \leftsquigarrow_F v$ if either $u \leftsquigarrow_T v$, or u is obtained by replacing a subword of the shape $w\top w$ in v, with w (for some guarded string w). Given a guarded string language L, we call *F-closure of L* the

guarded string language

$$C_F(L) \triangleq \{u \mid u \leftsquigarrow_F^* v \text{ for some } v \in L\}$$

C_F is a closure operator, but unlike C_T in the previous section, C_F is not a homomorphism. For instance, $C_F(\{\alpha a\alpha\} \cdot \{\alpha\top\alpha a\alpha\})$ contains $\alpha a\alpha$ while $C_F(\{\alpha a\alpha\}) \cdot C_F(\{\alpha\top\alpha a\alpha\})$ does not. Moreover, an elementary description of C_F requires more work than for C_T in the previous section.

Let E be the following function on guarded string languages

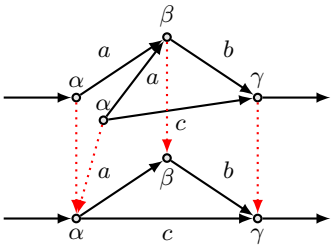
$$E(L) \triangleq \{w \mid \exists n,\ (w\top)^n w \in L\}$$

We shall prove that $C_F = E \circ C_T$, and that C_F can be characterised in terms of certain graph homomorphisms (Proposition 4.8 below).

Definition 4.3 (Graph, graph homomorphism)**.** A *graph* is a tuple $\langle V, E, l, \iota, o\rangle$, where V is a set of *vertices*, $E \subseteq V \times \Sigma \times V$ is a set of labelled edges, $l\colon V \to \mathsf{At}$ is a *node-labelling function*, and $\iota, o \in V$ are two distinguished vertices, respectively called *input* and *output*.

A *graph homomorphism* from the graph G to the graph H is a function from vertices of G to vertices of H that preserves node-labelling, labelled edges, input, and output. We write $H \lhd G$ when there exists a homomorphism from G to H.

The relation $\lhd$ on graphs is a preorder. We depict graphs as usual, using an unlabelled ingoing (resp. outgoing) arrow to indicate the input (resp. output); we use dotted red arrows to depict graph homomorphisms. For instance, we depict two finite connected graphs below, and a homomorphism between them:

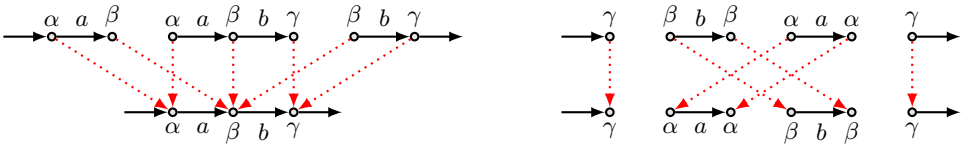


Definition 4.4 (Graph of a guarded string)**.** We associate to each guarded string u the graph $g(u)$ defined as follows:

- the vertices are the natural numbers smaller or equal to the length n of u;
- the ith vertex is labelled with the ith atom occurring in u;
- for $a \in \Sigma$ there is an a-labelled edge from i to $i+1$ if the i-th letter of u is a;
- the input is 0 and the output is n.

Graphs of guarded strings are rather simple: graphs as depicted above do not arise as graphs of guarded strings. For guarded strings not containing $\top$, they are just directed paths from the input to the output. For guarded strings containing $\top$, they are collections of (possibly empty) directed paths where the input is the starting-point of some path and the output is the end-point of some path. For example, the graphs of $\alpha a\beta b\gamma$ and $\alpha d\beta\top\alpha e\beta\top\gamma$ are depicted below:

Nevertheless, homomorphisms between graphs of guarded strings may be non-trivial. E.g., we have $g(\alpha a \beta b \gamma) \lhd g(\alpha a \beta \top \alpha a \beta b \gamma \top \beta b \gamma)$ and $g(\gamma \top \alpha a \alpha \top \beta b \beta \top \gamma) \lhd g(\gamma \top \beta b \beta \top \alpha a \alpha \top \gamma)$, as witnessed below:



In the sequel, we shall represent homomorphisms between graphs of guarded strings in a slightly more compact way, starting directly from the natural writing of the guarded strings, and using horizontal lines and shaded parallelograms to emphasise distinguished subwords and mappings between them. For instance, the above homomorphisms can be generalised to $g(u \diamond v) \lhd g(u \top u \diamond v \top v)$ and $g(\alpha \top u \top v \top \beta) \lhd g(\alpha \top v \top u \top \beta)$ for arbitrary guarded strings u, v and atoms α, β, which we can represent as follows:

Our main interest in graphs and homomorphisms comes from the following characterisation of the equational theory of REL. Without atoms, this characterisation appeared first in [BP15, Theorem 6], for the syntax of Kleene allegories. Its (trivial) extension to Kleene allegories with top then appeared in [Pou18, Theorem 16].

Theorem 4.5. *For all expressions e, f, we have:*

$$\mathsf{REL} \models e \leq f \iff \forall u \in [e],\ \exists v \in [f],\ g(u) \lhd g(v)$$

Proof. Cf. above references. That we need the theorem only in a small fragment here (without intersection and converse) does not seem to enable substantial simplifications. In particular, we still need to consider arbitrary graphs, and a variant of [AB95, Lemma 3] with top. That we deal with guarded string languages does not bring any difficulty, once we have the idea to label the graph vertices by atoms. We give a proof in Appendix A for the sake of completeness. □

Remark 4.6. For guarded strings u, v without top, we have $g(u) \lhd g(v)$ iff $u = v$. Therefore, for expressions e, f without top (whose languages only contain guarded strings without top), the above theorem reduces to $\mathsf{REL} \models e \leq f \iff [e] \subseteq [f]$, a standard variant of one of the equivalences recalled in the introduction [KS96, Theorem 6].

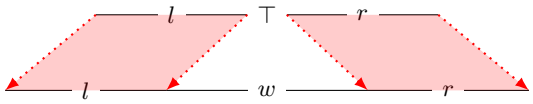
Thanks to Theorem 4.5, it suffices to relate homomorphisms between graphs of guarded strings to the notion of C_F-closure. We do so in the following lemma.

Lemma 4.7. *For all guarded strings u, v, the following are equivalent:*

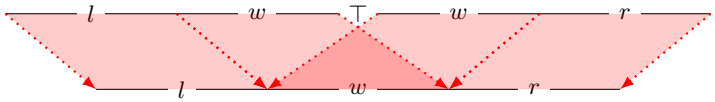
(i) $u \leftrightsquigarrow_F^* v$,
(ii) $g(u) \lhd g(v)$,
(iii) $u \in E(C_T\,\{v\})$.

Proof. We show (i) ⇒ (ii) ⇒ (iii) ⇒ (i). For the first implication, since $\lhd$ is a preorder, it suffices to show that $u \leftrightsquigarrow_F v$ entails $g(u) \lhd g(v)$. There are two cases to consider.

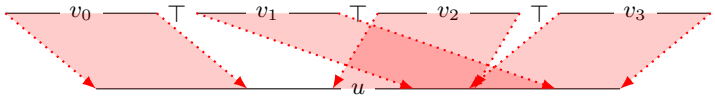
- either the rewriting rule associated to axiom (T) was used, i.e., $u = l \diamond w \diamond r$ and $v = l \top r$ for some guarded strings l, w, r. In that case we have the following homomorphism from the graph of v to the graph of u:



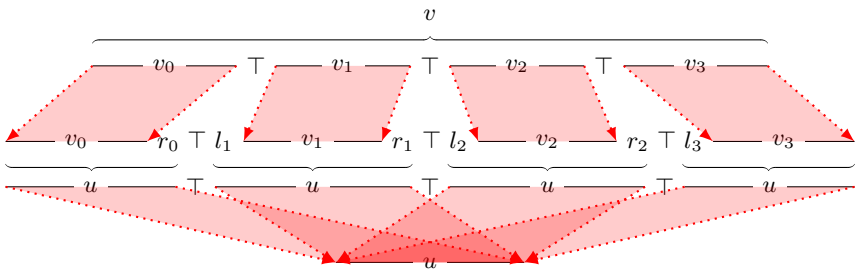
- or the rewriting rule associated to axiom (F) was used, i.e., $u = lwr$ and $v = lw \top wr$ for some words l, r and guarded string w. In that case we have the following homomorphism from the graph of v to the graph of u:



For the second implication, assume $g(u) \lhd g(v)$. Let n be the number of occurrences of $\top$ in v, and let $v_0, \dots, v_n$ be the top-free guarded strings such that $v = v_0 \top v_1 \top \cdots \top v_n$. Since they are top-free, those subwords must be mapped to subwords of u. For instance, when $n = 3$, the homomorphism may look as follows:



For all $0 \leq i \leq n$, let l_i, r_i be the words such that $u = l_i v_i r_i$. We have that l_0 and r_n must be the empty word since inputs and outputs must be preserved by homomorphisms. We have $(u\top)^n u \leftsquigarrow_T^n v$: we can obtain $(u\top)^n u$ from v by replacing the ith occurrence of $\top$ in v with the word $r_{i-1} \top l_i$, for $0 < i \leq n$. This suffices to conclude that $u \in E(C_T\{v\})$: we have proven (ii) $\Rightarrow$ (iii). As an example, when $n = 3$, the situation may be depicted as follows:



For the last implication, assume that $u \in E(C_T\{v\})$. There exists n such that $(u\top)^n u \leftsquigarrow_T^* v$, and thus in particular $(u\top)^n u \leftsquigarrow_F^* v$. Finally observe that $u \leftsquigarrow_F^n (u\top)^n u$ using n rewriting steps using (F), so that we can conclude by transitivity: $u \leftsquigarrow_F^n (u\top)^n u \leftsquigarrow_F^* v$. □

The above lemma has two important immediate consequences. First we have the announced factorisation of the closure C_F, and second, combined with Theorem 4.5, we obtain a characterisation of the equational theory of REL in terms of closed languages:

Proposition 4.8. *We have $C_F = E \circ C_T$.*

Proposition 4.9. *For all expressions e, f, we have:*

$$\mathsf{REL} \models e = f \iff C_F[e] = C_F[f]$$

Proof. For all e, f, we have:

$$\begin{aligned} \mathsf{REL} \models e \leq f &\iff \forall u \in [e],\ \exists v \in [f],\ g(u) \lhd g(v) && \text{(by Theorem 4.5)} \\ &\iff [e] \subseteq C_F[f] && \text{(by Lemma 4.7)} \end{aligned}$$

The initial statement follows by antisymmetry and the fact that C_F is a closure (so that for all languages L, K, $L \subseteq C_F(K)$ iff $C_F(L) \subseteq C_F(K)$). □

4.2. **Completeness w.r.t. closed guarded string languages.** It remains to show that KAT_F is complete w.r.t. the previous closed language interpretation ($C_F[\cdot]$). We use reductions in order to do so: we find a counterpart to the function r from Section 3 (Definition 3.4), for the F-closure rather than the T-closure. By Proposition 4.8, and since we already have the function r for T-closure, it actually suffices to find a function s that corresponds to the function E, i.e., such that for all expressions e, $s(e)$ is an expression whose language is $E[e]$ and such that $\mathsf{KAT}_F \vdash e = s(e)$.

4.2.1. *Interlude: monoids and the square root of a language.* Before defining the aforementioned reduction s for E, let us consider an exercise about plain regular languages. Define the *square root of a language* L as follows:

$$\sqrt{L} \triangleq \left\{ w \mid w^2 \in L \right\}$$

How to prove that this operation preserves regularity (i.e., if L is regular, then so is $\sqrt{L}$)? This is not obvious with automata-based techniques, but there is a simple and elegant solution using the characterisation of regular languages as those recognised by finite monoids [Eil74, Sak09].

Let us recall the corresponding definitions. A *monoid* is a tuple $\langle M, \cdot, 1 \rangle$ where M is a set and $\cdot$ is an associative binary operation on M with 1 as neutral element; it is *finite* when M is so. Words on Σ form a monoid, in fact the free monoid over Σ. A language L is *recognised by a finite monoid* M if there exists a homomorphism h from words to M and a subset P of M such that $L = h^{-1}(P)$ (i.e., for all words w, $w \in L$ iff $h(w) \in P$).

Proposition 4.10. *A language is regular iff it is recognised by a finite monoid.*

This result is entirely standard; we recall a proof below which will be helpful later to understand our construction on guarded string languages, and its complexity.

Proof.

- Given a finite non-deterministic automaton for a language L, the binary relations on its state-space form a finite monoid. Consider the unique homomorphism h mapping a letter to its transition relation; this homomorphism maps a word u to the relation containing all pairs of states that can be related by a u-labelled path in the automaton. Let P consist of all relations containing at least one pair of an initial state and a final state. We have $L = h^{-1}(P)$ by construction.
- Given a finite monoid $\langle M, \cdot, 1 \rangle$, a homomorphism h and a subset P such that $L = h^{-1}(P)$, we construct a finite deterministic automaton for L as follows: states are elements of the monoid; the initial state is the neutral element 1; the transition function maps a state x and a letter a to the state $x \cdot h(a)$; accepting states are the elements of P. □

Let us also recall the following basic property:

Lemma 4.11. *Let x, y be elements of a monoid and let h be a homomorphism from words to that monoid. We have the following language inclusion:*

$$h^{-1}(x) \cdot h^{-1}(y) \subseteq h^{-1}(x \cdot y)$$

Proof. A word in the left-hand side has shape uv for some words u, v such that $h(u) = x$ and $h(v) = y$. We then check that $h(uv) = h(u) \cdot h(v) = x \cdot y$, as required. □

To solve the exercise, let us generalise the previous square root function to subsets of arbitrary monoids. Given a subset P of a monoid, let $\sqrt{P}$ be the following subset:

$$\sqrt{P} \triangleq \{x \mid x^2 \in P\}$$

Proposition 4.12. *If $L = h^{-1}(P)$, then $\sqrt{L} = h^{-1}(\sqrt{P})$*

Proof. For all words w, we have

$$\begin{aligned} w \in \sqrt{L} &\iff w^2 \in L \\ &\iff h(w^2) \in P \\ &\iff h(w)^2 \in P && (h \text{ is a homomorphism}) \\ &\iff h(w) \in \sqrt{P} \\ &\iff w \in h^{-1}(\sqrt{P}) \end{aligned}$$

□

Thus, if L is regular, so is $\sqrt{L}$: we have solved our exercise. In particular, we can obtain a square root function on regular expressions such that $[\![\sqrt{e}]\!] = \sqrt{[\![e]\!]}$ for all e.

Adapting the above idea, we will obtain a function s such that $[s(e)] = E[e]$ for all expressions e. This does not explain how to show $\mathsf{KAT}_F \vdash s(e) \leq e$, however. We first show how to do so with our square root example. There, the counterpart to axiom (F) is the inequation $x \leq x^2$ (S). Accordingly, let KA_S denote the union of KA and (S).

Proposition 4.13. *For all regular expressions e, we have $\mathsf{KA}_S \vdash \sqrt{e} \leq e$.*

Proof. Since $[\![e]\!]$ is regular, we have $[\![e]\!] = h^{-1}(P) = \bigcup_{p \in P} h^{-1}(p)$ for some homomorphism h and subset P of a finite monoid. For all elements x in that monoid, let e_x be a regular expression for the language $h^{-1}(x)$ (this language being regular by Proposition 4.10).

The previous language equation can be rewritten as $[\![e]\!] = [\![\sum_{p \in P} e_p]\!]$. Similarly, we have $[\![\sqrt{e}]\!] = \sqrt{[\![e]\!]} = [\![\sum_{p \in \sqrt{P}} e_p]\!] = [\![\sum_{p^2 \in P} e_p]\!]$. By completeness of KA and then using axiom (S), we deduce

$$\mathsf{KA}_S \vdash \sqrt{e} = \sum_{p^2 \in P} e_p \leq \sum_{p^2 \in P} e_p^2$$

Therefore, to prove the announced statement, it suffices to show that $\mathsf{KA}_S \vdash e_p^2 \leq e$ whenever $p^2 \in P$. This follows by completeness of KA, since we have

$$[\![e_p^2]\!] = [\![e_p]\!]^2 = h^{-1}(p)^2 \subseteq h^{-1}(p^2) = [\![e_{p^2}]\!] \subseteq [\![e]\!]$$

(Where we use Lemma 4.11 for the first inclusion, and $p^2 \in P$ for the second one.) □

4.2.2. *Monoids for guarded string languages.* On regular expressions with top but without atoms, one can easily adapt the previous idea to obtain a reduction for E. Indeed, if $L = h^{-1}(P)$, then $E(L) = h^{-1}(P')$ for P' defined as follows:

$$P' = \{x \mid \exists n, (x \cdot h(\top))^n \cdot x \in P\}$$

(Note that P' can be computed when the monoid is finite: there are only finitely many powers of $x \cdot h(\top)$.) This approach gives a simpler alternative to the automata construction we defined in [PW22, Section 4.2].

However, to deal with full expressions and guarded string languages, we first need to extend the theory of recognition by finite monoids to such languages.

To do so, we rely on the presentation of guarded strings as elements of $(\mathsf{At} \times \Sigma_\top)^* \times \mathsf{At}$. A word over $\mathsf{At} \times \Sigma_\top$ can be seen as a word over $\Sigma_\top \uplus \mathsf{At}$ (e.g., $(\alpha, a)(\beta, b) = \alpha a \beta b$), and when u is such a word and α is an atom, $u\alpha$ is a guarded string. All guarded strings can be decomposed in this way (uniquely).

Every regular expression $\mathbf{e}$ over $\mathsf{At} \times \Sigma_\top$ can be seen as an expression $\underline{\mathbf{e}}$ by replacing each letter (α, a) by the expression $\alpha \cdot a$. We call *clean* the expressions of the form $\underline{\mathbf{e}}$. The languages of such expressions are almost guarded string languages: their words only miss the final atom.

Fact 4.14. For all clean expressions e and atoms α, we have $[e \cdot \alpha] = [\![e]\!] \cdot \{\alpha\}$.

Definition 4.15. A *recogniser* is a tuple $\langle M, h, P\rangle$ where M is a monoid, h is a homomorphism from $(\mathsf{At} \times \Sigma_\top)^*$ to M, and P is a subset of $M \times \mathsf{At}$; it is *finite* when M is so. Given such a recogniser, we define the following guarded string language:

$$h^{-1}(P) \triangleq \{u\alpha \mid P(h(u), \alpha)\}$$

We also keep the notation from Section 4.2.1: when x is an element of the monoid, $h^{-1}(x) = \{u \mid h(u) = x\}$ is a language over $\mathsf{At} \times \Sigma_\top$. In particular, we have

$$h^{-1}(P) = \bigcup_{P(p,\alpha)} h^{-1}(p) \cdot \{\alpha\}$$

Proposition 4.16. *For all expressions e, there exists a finite recogniser $\langle M, h, P\rangle$ such that $[e] = h^{-1}(P)$.*

Proof. First we compute a non-deterministic finite automaton over the alphabet $\Sigma_\top \uplus \mathsf{At}$, for $[\![e]\!]$. Such an automaton is a tuple $\langle S, I, \Delta, F\rangle$ where S is a finite set of states, $I, F \subseteq S$ are the initial and accepting states, respectively, and Δ is the transition relation, seen as a map from $\Sigma_\top \uplus \mathsf{At}$ to relations on S. By construction, we have

$$x_1 \dots x_n \in [\![e]\!] \iff \Delta(x_1) \cdot \dots \cdot \Delta(x_n) \cap I \times F \neq \emptyset$$

(For all words $x_1 \dots x_n$ over $\Sigma_\top \uplus \mathsf{At}$.) Recall the function gs extracting the guarded strings of a language, such that $[e] = \mathsf{gs}[\![e]\!]$. We deduce that for all guarded strings $\alpha_0 a_1 \dots a_{n-1} \alpha_n$, we have

$$\alpha_0 a_1 \dots a_{n-1} \alpha_n \in [e] \iff \Delta(\alpha_0)^* \cdot \Delta(a_1) \cdot \dots \cdot \Delta(a_{n-1}) \cdot \Delta(\alpha_n)^* \cap I \times F \neq \emptyset$$

Accordingly, we construct a finite recogniser as follows: M is the monoid of relations on S; h is the unique homomorphism such that $h(\alpha, a) = \Delta(\alpha)^* \cdot \Delta(a)$ for all atoms α and $a \in \Sigma_\top$; and $P(R, \alpha)$ holds if $R \cdot \Delta(\alpha)^* \cap I \times F \neq \emptyset$. □

The converse also holds: one can associate an expression to every finite recogniser. We need a slightly stronger statement in the sequel:

Proposition 4.17. *For all finite recognisers $\langle M, h, P\rangle$, there are clean expressions $(e_x)_{x\in M}$ such that for all $x \in M$, $[\![e_x]\!] = h^{-1}(x)$. It follows that*

$$h^{-1}(P) = \left[\sum_{P(p,\alpha)} e_p \cdot \alpha\right]$$

Proof. For each $x \in M$, let $\mathbf{e}_x$ be a regular expression over $\mathsf{At} \times \Sigma_\top$ for $h^{-1}(x)$ and set $e_x \triangleq \underline{\mathbf{e}_x}$. We deduce via Fact 4.14 that

$$h^{-1}(P) = \bigcup_{P(p,\alpha)} h^{-1}(p)\cdot\{\alpha\} = \bigcup_{P(p,\alpha)} [\![e_p]\!]\cdot\{\alpha\} = \bigcup_{P(p,\alpha)} [e_p\cdot\alpha] = \left[\sum_{P(p,\alpha)} e_p\cdot\alpha\right] \qquad \square$$

4.2.3. *Reduction for E and completeness.* Now that the monoid machinery is set up for guarded string languages, we can show that the function E preserves regularity.

Proposition 4.18. *Let $\langle M, h, P\rangle$ be a recogniser and define P' as follows:*

$$P' \triangleq \{(x,\alpha) \mid \exists n \in \mathbb{N},\ P((x\cdot h(\alpha,\top))^n \cdot x, \alpha)\}$$

We have $E(h^{-1}(P)) = h^{-1}(P')$.

Proof. For all guarded strings $u\alpha$, we have

$$\begin{aligned}
u\alpha \in E(h^{-1}(P)) &\iff \exists n\in\mathbb{N},\ (u\alpha\top)^n u\alpha \in h^{-1}(P)\\
&\iff \exists n\in\mathbb{N},\ P(h((u\alpha\top)^n u),\alpha)\\
&\iff \exists n\in\mathbb{N},\ P((h(u)\cdot h(\alpha,\top))^n\cdot h(u),\alpha) \qquad (h \text{ is a homomorphism})\\
&\iff P'(h(u),\alpha)\\
&\iff u\alpha \in h^{-1}(P')
\end{aligned}$$

$\square$

Fact 4.19. In the above proposition, when the recogniser is obtained from a transition monoid (Proposition 4.16), we have the following alternative presentation of P':

$$P'(X,\alpha) = P((X\cdot T_\alpha)^*\cdot X,\alpha) \quad \text{where} \quad T_\alpha \triangleq h(\alpha,\top) = \Delta(\alpha)^*\cdot\Delta(\top)$$

Proof. Because $\exists n,\ Y^n\cdot Z\cap I\times F\neq\emptyset$ iff $(\bigcup_n Y^n\cdot Z)\cap I\times F\neq\emptyset$ iff $(Y^*\cdot Z)\cap I\times F\neq\emptyset$. $\square$

We can finally define the reduction s. Given an expression e, first compute a finite recogniser $\langle M, h, P\rangle$ such that $[e] = h^{-1}(P)$ (Proposition 4.16), then update P into P' as in Proposition 4.18, and finally extract the expression $s(e)$ from $\langle M, h, P'\rangle$ (Proposition 4.17).

Proposition 4.20. *For all expressions e, we have*

(i) $[s(e)] = E[e]$, *and*
(ii) $\mathsf{KAT}_F \vdash e = s(e)$.

Proof. The first item follows by construction and the three previous propositions. For the second one, since $[e] \subseteq E[e] = [s(e)]$, we have $\mathsf{KAT} \vdash e \leq s(e)$ by completeness of KAT (‡), so that it suffices to show the other inequation.

Let $\langle M, h, P\rangle$ be the finite recogniser for $[e]$ used to construct $s(e)$ and let $(e_x)_{x\in M}$ be the expressions given by Proposition 4.17.

We have $[e] = [\sum_{P(p,\alpha)} e_p \cdot \alpha]$ and $[s(e)] = [\sum_{P'(p,\alpha)} e_p \cdot \alpha]$. By completeness of KAT (†), we deduce

$$\mathsf{KAT} \vdash s(e) = \sum_{P'(p,\alpha)} e_p \cdot \alpha$$

Therefore, it suffices to show that $\mathsf{KAT}_F \vdash e_p \cdot \alpha \leq e$ whenever $P'(p,\alpha)$. Accordingly, let n, p, α be such that $P((p \cdot h(\alpha, \top))^n \cdot p, \alpha)$. Set $t \triangleq h(\alpha, \top)$ and $q \triangleq (p \cdot t)^n \cdot p$; we have $P(q, \alpha)$. We derive

$$\begin{aligned} \mathsf{KAT}_F \vdash e_p \cdot \alpha &\leq (e_p \cdot \alpha \cdot \top)^n \cdot e_p \cdot \alpha && \text{(using axiom (F) } n \text{ times)} \\ &\leq e_q \cdot \alpha \\ &\leq e \end{aligned}$$

For the last two steps, we use KAT completeness (‡): we have

$$\begin{aligned} [(e_p \cdot \alpha \cdot \top)^n \cdot e_p \cdot \alpha] &= [\![(e_p \cdot \alpha \cdot \top)^n \cdot e_p]\!] \cdot \{\alpha\} && \text{(Fact 4.14: } (e_p \cdot \alpha \cdot \top)^n \cdot e_p \text{ is clean)} \\ &= ([\![e_p]\!] \cdot \{\alpha\top\})^n \cdot [\![e_p]\!] \cdot \{\alpha\} && ([\![\cdot]\!] \text{ is a homomorphism}) \\ &= (h^{-1}(p) \cdot \{\alpha\top\})^n \cdot h^{-1}(p) \cdot \{\alpha\} && \text{(definition of } e_p\text{—Proposition 4.17)} \\ &\subseteq (h^{-1}(p) \cdot h^{-1}(t))^n \cdot h^{-1}(p) \cdot \{\alpha\} && (h(\alpha, \top) = t) \\ &\subseteq h^{-1}((p \cdot t)^n \cdot p) \cdot \{\alpha\} && \text{(by Lemma 4.11)} \\ &= h^{-1}(q) \cdot \{\alpha\} && \text{(definition of } q) \\ &= [\![e_q]\!] \cdot \{\alpha\} && \text{(definition of } e_q\text{—Proposition 4.17)} \\ &= [e_q \cdot \alpha] && \text{(Fact 4.14)} \\ &\subseteq [e] && (P(q,\alpha)) \quad \square \end{aligned}$$

We finally combine all the above results to obtain our main theorem:

Theorem 4.21. *For all regular expressions with top e, f, we have*

$$\mathsf{REL} \models e = f \iff C_F[e] = C_F[f] \iff \mathsf{KAT}_F \vdash e = f$$

Proof. We have

$$\begin{aligned} & \mathsf{REL} \models e = f \\ \Leftrightarrow\ & C_F[e] = C_F[f] && \text{(Proposition 4.9)} \\ \Leftrightarrow\ & E(C_T[e]) = E(C_T[f]) && \text{(by Proposition 4.8)} \\ \Leftrightarrow\ & [s(r(e))] = [s(r(f))] && \text{(by Propositions 3.5(i) and 4.20(i))} \\ \Leftrightarrow\ & \mathsf{KAT} \vdash s(r(e)) = s(r(f)) && \text{(by completeness of } \mathsf{KAT} \text{ (†))} \\ \Rightarrow\ & \mathsf{KAT}_F \vdash e = f && \text{(by transitivity and Propositions 3.5(ii) and 4.20(ii))} \\ \Rightarrow\ & \mathsf{REL} \models e = f && \text{(soundness of } \mathsf{KAT}_F \text{ axioms w.r.t. } \mathsf{REL}) \quad \square \end{aligned}$$

The above proof follows the same strategy as the one for Theorem 3.6. Like there, the right-to-left implication of the second equivalence in the statement is an instance of [DKPP19, Theorem 2] (generalised to guarded string languages), and we use reductions only for the left-to-right part of this equivalence.

4.3. **Solution to the exercise from the introduction.** Recall the exercise ($\star$) from the introduction. Since this example does not involve tests, we work with KA rather than KAT, we identify words and guarded strings, and we simplify definitions according to Remark 2.1: there is only one atom. We first give a handcrafted solution, before illustrating how the previous construction works on this example.

4.3.1. *Handcrafted solution.* For a number $i \geq 2$, let us write i for the expression a^i. We have to prove the following inequation using KA and axiom (F):

$$3^* \leq 3^*\top 2^* + 2^*a\top 3^*$$

As a first hint, observe that $\mathsf{KA} \vdash i^* \leq j^*$ when i is a multiple of j.
As a second hint, let us prove:

$$6^* \leq 3^*\top 2^*$$

Indeed, we have $6^* \leq 6^*\top 6^*$ by axiom (F), and both $6^* \leq 3^*$ and $6^* \leq 2^*$ since 6 is a multiple of both 3 and 2.
Similarly, we can also prove:

$$6^*3 \leq 2^*a\top 3^*$$

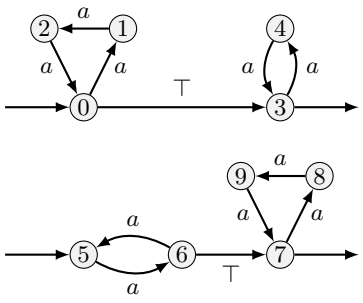
Indeed, we have $6^*3 \leq 6^*3\top 6^*3$ by axiom (F), and then $6^*3 \leq 2^*2a \leq 2^*a$ and $6^*3 \leq 3^*3 \leq 3^*$ by basic KA reasoning.

Finally observe that $[3^*] \subseteq [6^* + 6^*3]$: every multiple of 3 is also either even or odd. This suffices to conclude by KA completeness:

$$3^* \leq 6^* + 6^*3 \leq 3^*\top 2^* + 2^*a\top 3^*$$

4.3.2. *Computed solution.* Now let us see how this solution can be obtained via our generic construction. We purposely skip the reduction r for axiom (T), which we want to avoid. Set $e \triangleq 3^*\top 2^* + 2^*a\top 3^*$. We will get $\mathsf{KA}_F \vdash 3^* \leq s(e) = e$ using KA completeness for the first step (after checking that $[3^*] \subseteq E[e] = [s(e)]$), and Proposition 4.20(ii) for the second step.

In order to compute $s(e)$, consider the following automaton for $[e]$:



The associated monoid is huge: it has 2^{10^2} elements; however, only the elements in the image of the homomorphism are relevant, and we shall see that it suffices to look at six of them.

Let $A \triangleq \Delta(a)$ and $T \triangleq \Delta(\top)$ be the transition relations for a and $\top$. Looking at A and T as 01-matrices, T only contains two non-zero entries, at positions $(0, 3)$ and $(6, 7)$, and A

can be presented as a block-diagonal matrix:

$$A \triangleq \begin{pmatrix} M & & & \\ & N & & \\ & & N & \\ & & & M \end{pmatrix} \quad \text{where} \quad M \triangleq \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \quad \text{and} \quad N \triangleq \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

We have $M^3 = 1$ and $N^2 = 1$, so that $A^6 = 1$ and there are only six distinct matrices of the shape A^i. The homomorphism $h = \hat{\Delta}$ maps words over $\{a, \top\}$ to 10x10 matrices. Words of the shape a^i are mapped to A^i, words with at least two occurrences of $\top$ are mapped to the zero matrix, and words of the shape $a^i \top a^j$ are mapped to $A^i T A^j$. Therefore, there are at most 6+1+36 elements in the image of h.

Let us now compute the predicate P on these elements (here this predicate takes a single argument since there is only one atom). We write $i \equiv j[k]$ for i equals j modulo k):

$$\begin{cases} P(A^i) = \text{false} \\ P(0) = \text{false} \\ P(A^i T A^j) = (i \equiv 0[3] \wedge j \equiv 0[2]) \vee (i \equiv 1[2] \wedge j \equiv 0[3]) \end{cases}$$

Now observe that $(XT)^*X = X + XTX$ when $X = A^i$, and $(XT)^*X = X$ in the other cases ($X = 0$ or $X = A^i T A^j$). By Fact 4.19, we have $P'(A^i) = P(A^i + A^i T A^i)$, and $P'(X) = P(X)$ in the other cases. Further simplifying $P'(A^i)$, we get:

$$\begin{aligned} P'(A^i) &= P(A^i + A^i T A^i) \\ &= \text{false} \vee (i \equiv 0[3] \wedge i \equiv 0[2]) \vee (i \equiv 1[2] \wedge i \equiv 0[3]) \\ &= i \equiv 0[3] \wedge (i \equiv 0[2] \vee i \equiv 1[2]) \\ &= i \equiv 0[3] \end{aligned}$$

In other words, with respect to the predicate P, there are only two elements in the image of h that are added to obtain P': A^0 and A^3.

We finally observe that $h^{-1}(A^0) = [6^*]$ and $h^{-1}(A^3) = [6^*3]$, so that

$$[s(e)] = h^{-1}(P') = h^{-1}(P) \cup h^{-1}(A^0) \cup h^{-1}(A^3) = [e + 6^* + 6^*3]$$

4.4. **A PSpace algorithm.** Like for Theorem 3.6, the proof of Theorem 4.21 leads to an algorithm for deciding the equational theory of KAT_F. Indeed, we have $\mathsf{KAT}_F \vdash e = f$ iff $[s(r(e))] = [s(r(f))]$, so that it suffices to be able to compare the latter guarded string languages. This is possible since the functions r and s are computable. However, while the function r is linear, the function s is costly: extracting a regular expression from a monoid (or similarly from a finite automaton) is exponential in general.

To obtain a PSpace algorithm, we avoid computing s and work directly with recognisers. Indeed, given an expression e of size n, our constructions define a finite recogniser as in Figure 1. This gives us a monoid for $E[e]$ whose elements are binary relations over $O(n)$ states[4]. In other words, elements are square 01-matrices of dimension $O(n) \times O(n)$. Those elements can be stored in quadratic space, and the various operations of the recogniser can be computed in polynomial time:

- the monoid product is nothing but matrix multiplication;

[4]Assuming a regular-expression-to-automata function producing non-deterministic finite automata with linearly many states, as is usually the case [Tho68, Ant96].

```
// inputs an expression e; outputs a recogniser for E[e]
⟨X, I, Δ, F⟩ := non-deterministic finite automaton for [e]
M := ⟨P(X²), ·, 1⟩
h(α, a) := Δ(α)* · Δ(a)
P(R, α) := (R · Δ(α)* · Δ(⊤))* · R · Δ(α)* ∩ I×F ≠ ∅
return ⟨M, h, P⟩
```

Figure 1: Recogniser for a language of the form $E[e]$.

```
// inputs two expressions e, f; outputs false iff E[r(e)] ≠ E[r(f)]
⟨M, h, P⟩ := recogniser for E[r(e)]
⟨N, g, Q⟩ := recogniser for E[r(f)]
x := 1_M
y := 1_N
while true do
  guess α ∈ At
  if P(x, α) ≠ Q(y, α) then return false
  guess a ∈ Σ⊤
  x := x ·_M h(α, a)
  y := y ·_N g(α, a)
done
```

Figure 2: Non-deterministic PSpace algorithm for the equational theory of KAT_F.

- calling the homomorphism h on a pair (α, a) requires a matrix multiplication and a reflexive-transitive closure;
- testing whether a pair (R, α) is accepted requires three multiplications and two reflexive-transitive closures. (Note that the formula we use for P in Figure 1 comes from Fact 4.19.)

Putting everything together, we obtain the algorithm in Figure 2. This algorithm is non-deterministic: it progressively guesses a potential counter-example—a guarded string—and checks whether it is indeed a counter-example using recognisers for closed languages of guarded strings as deterministic guarded string automata. This algorithm requires quadratic space: it stores only the two 01-matrices x and y, whose respective dimensions are linear in the sizes of $r(e)$ and $r(f)$, and thus e and f.

It may seem surprising that this algorithm has an endless loop and never returns *true*. Still, we can turn it into a (deterministic, terminating) PSpace algorithm by Savitch' theorem [Sav70]. Intuitively, we can explore all non-deterministic choices and halt returning *true* when all configurations (i.e., pairs $\langle x, y\rangle$ of 01-matrices) have been visited and no counter-example was found.

The equational theory of KAT_F contains that of KA, which amounts to language equivalence of regular expressions, which is PSpace-hard [MS72, Lemma 2.3][HRS76, Proposition 2.4]. Therefore we deduce:

Theorem 4.22. *The equational theory of* KAT_F *is* PSpace*-complete.*

5. Relations with a greatest element

A *generalised $S_\top$-algebra of relations* is an S-subalgebra A of an algebra of relations such that A has a greatest element, seen as an $S_\top$-algebra by using this greatest element for the constant $\top$. We write REL' for the class of all generalised $S_\top$-algebras of relations.

Intuitively, REL' consists of models of binary relations where $\top$ is not necessarily the full relation, only a greatest element. As an example, consider relations R over the natural numbers such that $i \leq j$ whenever $i\,R\,j$. Those form an S-algebra with greatest element the order relation $\leq$ itself, which is not the full relation.

In the literature, REL' is sometimes preferred over REL because it is closed under taking subalgebras and products, and actually forms a quasivariety [AM11]. (In contrast, it is not clear whether REL is closed under products: the two obvious ways of embedding a pair of relations into a new relation fail to preserve either union or top—REL as defined here is not closed under taking subalgebras either, but defining it in such a way would not change the results from the present paper.)

The equational theory of REL' differs from that of REL. For instance, the previous example of ordered relations shows that $\mathsf{REL}' \not\models x \leq x \cdot \top \cdot x$. Indeed, for $x = \{\langle 0, 1\rangle\}$, $x \cdot \top \cdot x$ is empty since $\top$ does not relate 1 to 0.

We show below that the equational theory of REL' actually coincides with that of GSL, and can thus be axiomatised by KAT_T.

Proposition 5.1. *Every member of* GSL *embeds into a member of* REL'.

Proof. We adapt the technique used by Pratt for Kleene algebras (without top) [Pra80, third page] and later reused by Kozen and Smith for Kleene algebras with tests [KS96, Lemma 5]. For a set X, let $M(X)$ be the set of relations R on $\mathcal{GS}_X$ such that for all guarded strings u, v, u is a prefix of v whenever $u\,R\,v$. The S-operations on relations restrict to $M(X)$, so that $M(X)$ is an S-algebra, and setting $\top \triangleq \{\langle u, u \diamond v\rangle \mid u, v \in \mathcal{GS}_X\}$ turns it into a member of REL'. We embed the member $\mathcal{P}(\mathcal{GS}_X)$ of GSL into $M(X)$ as follows:

$$\begin{aligned} \iota \colon \mathcal{P}(\mathcal{GS}_X) &\to M(X) \\ L &\mapsto \{\langle u, u \diamond v\rangle \mid u \in \mathcal{GS}_X,\ v \in L\} \end{aligned}$$

The function ι is easily shown to be an $S_\top$-algebra homomorphism, and it is injective (since, e.g., $L = \{\alpha u \mid \langle \alpha, \alpha u\rangle \in \iota(L)\}$). □

Note that it is crucial that we consider REL' rather than REL here: the above construction would not give an $S_\top$-algebra homomorphism if we were not restricting to relations of a certain shape: $\top$ would not be preserved.

Corollary 5.2. *For all expressions, we have*

$$\mathsf{GSL} \models e = f \iff \mathsf{REL}' \models e = f \iff \mathsf{KAT}_T \vdash e = f$$

Proof. That $\mathsf{REL}' \models e = f$ entails $\mathsf{GSL} \models e = f$ is a direct consequence of Proposition 5.1. That $\mathsf{KAT}_T \vdash e = f$ entails $\mathsf{REL}' \models e = f$ follows from the soundness of KAT_T axioms w.r.t. REL'. We conclude by Theorem 3.6. □

Similarly to REL', we can define a class GSL' of $S_\top$-algebras which is closed under taking subalgebras and where $\top$ is not necessarily the full language. However, unlike with REL' and REL, the equational theory of GSL' coincides with that of GSL (and REL'). Indeed the axioms of KAT_T remain sound for GSL'.

6. Conclusion

We have proved completeness of two axiomatic systems about regular expressions with tests and top, KAT_T and KAT_F, with respect to guarded string language models and relational models, respectively. We have established that the corresponding equational theories are PSpace-complete, and that they can be reconciled by allowing relational models where top is only a maximal element, not necessarily the full relation.

For KAT_F, we have proved a graph-theoretical characterisation of the equational theory of binary relations, we have established a relationship between this graph-theoretical characterisation and a notion of closed guarded string language, and we have used an extension of the theory of finite monoid recognition for guarded string languages.

Related work. Zhang et al. gave a completeness result for KAT_T, in terms of guarded string languages [ZdAG22, Theorem 9]. They observed that this axiomatisation is incomplete for REL, that it does not suffice to properly express *incorrectness triples*, and they left the existence of a complete axiomatisation for relational models open. Our Theorem 4.21 gives a positive answer to this question.

For the theory KAT_T, the main completeness results of Zhang et al. [ZdAG22, Theorems 7 and 9] are wrong: the model of guarded strings they designed equates too many expressions (namely, Σ^* and $\top$—see Remark 3.7). Our Theorem 3.6 uses a slightly different language model and yields a linear reduction from KAT_T to KAT, so that, e.g., [ZdAG22, Theorem 10] about the complexity of KAT_T remains true.

Zhang et al. also gave a completeness result w.r.t. generalised relational models [ZdAG22, Theorem 8]. Their proof is problematic because it relies on their Theorem 7, but the key idea remains valid: adapting Pratt's trick to embed language models into relational ones. We use the very same technique to obtain Corollary 5.2.

We recently proved a completeness result for KAT_F [PRW24, Section 7], w.r.t. a notion of closed language defined differently than in the present work. There the emphasis is on modularity, complexity aspects are not considered, and KAT with a top element is an example among others. The closed language model defined there is most probably equivalent to the one we use in the present paper (using arguments likes the one developed in [PRW24, Appendix C] for plain KAT, which we would like to generalise in the future). The two completeness proofs are rather different. The present one is more direct, uses guarded strings and finite monoids, and yields a PSpace algorithm. In contrast, the one in [PRW24] avoids guarded strings but requires more general results about Kleene algebra with hypotheses, and does not give any reasonable algorithm. Our characterisations of the equational theory of REL (Theorem 4.5, Proposition 4.9) also lie out of the scope of [PRW24].

Future work. A Hoare triple $\{\alpha\}\, e\, \{\beta\}$ for partial correctness can be encoded in KAT as an equation $\alpha \cdot e \cdot \neg\beta = 0$ [Koz00]. Since hypotheses of the more general shape $e = 0$ can be incorporated into the equational theory of KAT [Coh94, HK02], one can automate reasoning about partial correctness [Pou13].

Zhang et al. [ZdAG22] have shown how to encode an incorrectness triple $[\alpha]e[\beta]$ as an inequation $\beta \leq \top \cdot \alpha \cdot e$. A natural question is whether such hypotheses can also be eliminated in KAT_F, in order to automate reasoning about incorrectness triples. The modular tools we developed in [PRW21, PRW24] could prove useful, provided we find a way to extract efficient algorithms from the resulting reductions.

Acknowledgements. We would like to thank Paul Brunet, Amina Doumane, and Jurriaan Rot for the discussions that eventually led to this work, and the CONCUR'22 reviewers for all their comments. We also thank Denis Kuperberg for showing us how the roots of a regular language could be computed via finite monoids, which is the key idea that led to the construction in Section 4.2.

References

[AB95] Hajnal Andréka and Dmitry A. Bredikhin. The equational theory of union-free algebras of relations. *Algebra Universalis*, 33(4):516–532, 1995. doi:10.1007/BF01225472.

[AFG+14] Carolyn Jane Anderson, Nate Foster, Arjun Guha, Jean-Baptiste Jeannin, Dexter Kozen, Cole Schlesinger, and David Walker. Netkat: semantic foundations for networks. In Suresh Jagannathan and Peter Sewell, editors, *POPL*, pages 113–126. ACM, 2014. doi:10.1145/2535838.2535862.

[AM11] Hajnal Andréka and Szabolcs Mikulás. Axiomatizability of positive algebras of binary relations. *Algebra Universalis*, 66(1):7–34, 2011. doi:10.1007/s00012-011-0142-3.

[AMN11] Hajnal Andréka, Szabolcs Mikulás, and István Németi. The equational theory of Kleene lattices. *Theoretical Computer Science*, 412(52):7099–7108, 2011. doi:10.1016/j.tcs.2011.09.024.

[Ant96] Valentin M. Antimirov. Partial derivatives of regular expressions and finite automaton constructions. *Theoretical Computer Science*, 155(2):291–319, 1996. doi:10.1016/0304-3975(95)00182-4.

[BÉS95] Stephen L. Bloom, Zoltán Ésik, and Gheorghe Stefanescu. Notes on equational theories of relations. *Algebra Universalis*, 33(1):98–126, 1995. doi:10.1007/BF01190768.

[Bof90] Maurice Boffa. Une remarque sur les systèmes complets d'identités rationnelles. *Informatique Théorique et Applications*, 24:419–428, 1990. URL: http://archive.numdam.org/article/ITA_1990__24_4_419_0.pdf.

[Bof95] Maurice Boffa. Une condition impliquant toutes les identités rationnelles. *Informatique Théorique et Applications*, 29(6):515–518, 1995. URL: http://www.numdam.org/article/ITA_1995__29_6_515_0.pdf.

[BP15] Paul Brunet and Damien Pous. Petri automata for Kleene allegories. In *LICS*, pages 68–79. ACM, 2015. doi:10.1109/LICS.2015.17.

[BP16] Paul Brunet and Damien Pous. Algorithms for Kleene algebra with converse. *Journal of Logical and Algebraic Methods in Programming*, 85(4):574–594, 2016. doi:10.1016/j.jlamp.2015.07.005.

[CKS96] Ernie Cohen, Dexter Kozen, and Frederick Smith. The complexity of Kleene algebra with tests. Technical Report TR96-1598, CS Dpt., Cornell University, 1996. URL: http://www.cs.cornell.edu/~kozen/papers/ckat.pdf.

[Coh94] Ernie Cohen. Hypotheses in Kleene algebra. Technical report, Bellcore, Morristown, N.J., 1994. URL: http://www.researchgate.net/publication/2648968_Hypotheses_in_Kleene_Algebra.

[Con71] John Horton Conway. *Regular Algebra and Finite Machines*. Chapman and Hall mathematics series. Chapman and Hall, 1971. URL: https://books.google.nl/books?id=1KAXc5TpEV8C.

[DDP18] Anupam Das, Amina Doumane, and Damien Pous. Left-handed completeness for Kleene algebra, via cyclic proofs. In *LPAR*, volume 57 of *EPiC Series in Computing*, pages 271–289. Easychair, 2018. doi:10.29007/hzq3.

[DKPP19] Amina Doumane, Denis Kuperberg, Pierre Pradic, and Damien Pous. Kleene algebra with hypotheses. In *FoSSaCS*, volume 11425 of *LNCS*, pages 207–223. Springer, 2019. doi:10.1007/978-3-030-17127-8_12.

[DP18] Amina Doumane and Damien Pous. Completeness for identity-free Kleene lattices. In *CONCUR*, volume 118 of *LIPIcs*, pages 18:1–18:17. Schloss Dagstuhl, 2018. doi:10.4230/LIPIcs.CONCUR.2018.18.

[ÉB95] Zoltán Ésik and Laszlo Bernátsky. Equational properties of Kleene algebras of relations with conversion. *Theoretical Computer Science*, 137(2):237–251, 1995. doi:10.1016/0304-3975(94)00041-G.

[Eil74] Samuel Eilenberg. *Automata, Languages, and Machines*. Automata, Languages, and Machines. Academic Press, 1974. URL: https://books.google.fr/books?id=VaFyxwEACAAJ.

[FS90] Peter Freyd and Andre Scedrov. *Categories, Allegories*. North Holland, 1990.

[HK02] Christopher Hardin and Dexter Kozen. On the elimination of hypotheses in Kleene algebra with tests. Technical Report TR2002-1879, CS Dpt., Cornell University, October 2002. URL: http://hdl.handle.net/1813/5855.

[HRS76] Harry B. Hunt, Daniel J. Rosenkrantz, and Thomas G. Szymanski. On the equivalence, containment, and covering problems for the regular and context-free languages. *Journal of Computer and System Sciences*, 12(2):222–268, 1976. doi:10.1016/S0022-0000(76)80038-4.

[KBS+20] Tobias Kappé, Paul Brunet, Alexandra Silva, Jana Wagemaker, and Fabio Zanasi. Concurrent Kleene algebra with observations: from hypotheses to completeness. In *FoSSaCS*, volume 12077 of *LNCS*, pages 381–400. Springer, 2020. doi:10.1007/978-3-030-45231-5_20.

[Kle56] Stephen Cole Kleene. Representation of events in nerve nets and finite automata. In *Automata Studies*, pages 3–41. Princeton University Press, 1956. URL: http://www.rand.org/pubs/research_memoranda/2008/RM704.pdf.

[KN12] Alexander Krauss and Tobias Nipkow. Proof pearl: Regular expression equivalence and relation algebra. *Journal of Automated Reasoning*, 49(1):95–106, 2012. doi:10.1007/s10817-011-9223-4.

[Koz91] Dexter Kozen. A completeness theorem for Kleene Algebras and the algebra of regular events. In *LICS*, pages 214–225. IEEE Computer Society, 1991. doi:10.1109/LICS.1991.151646.

[Koz97] Dexter Kozen. Kleene algebra with tests. *ACM Transactions on Programming Languages and Systems*, 19(3):427–443, May 1997. doi:10.1145/256167.256195.

[Koz00] Dexter Kozen. On Hoare logic and Kleene algebra with tests. *ACM Transactions on Computational Logic*, 1(1):60–76, 2000. doi:10.1145/343369.343378.

[Koz17] Dexter Kozen. On the coalgebraic theory of Kleene algebra with tests. In Can Başkent, Lawrence S. Moss, and Ramaswamy Ramanujam, editors, *Rohit Parikh on Logic, Language and Society*, volume 11 of *Outstanding Contributions to Logic*, pages 279–298. Springer, March 2017.

[KP00] Dexter Kozen and Maria-Christina Patron. Certification of compiler optimizations using Kleene algebra with tests. In *CL2000*, volume 1861 of *LNAI*, pages 568–582. Springer, 2000. doi:10.1007/3-540-44957-4_38.

[Kro91] Daniel Krob. Complete systems of B-rational identities. *Theoretical Computer Science*, 89(2):207–343, 1991. doi:10.1016/0304-3975(91)90395-I.

[KS96] Dexter Kozen and Frederick Smith. Kleene algebra with tests: Completeness and decidability. In *CSL*, volume 1258 of *LNCS*, pages 244–259. Springer, September 1996. doi:10.1007/3-540-63172-0_43.

[KS12] Dexter Kozen and Alexandra Silva. Left-handed completeness. In *RAMiCS*, volume 7560 of *LNCS*, pages 162–178. Springer, 2012. doi:10.1007/978-3-642-33314-9_11.

[Mam17] Konstantinos Mamouras. Equational theories of abnormal termination based on Kleene algebra. In *FoSSaCS*, volume 10203 of *LNCS*, pages 88–105. Springer, 2017. doi:10.1007/978-3-662-54458-7_6.

[MS72] Albert R. Meyer and Larry J. Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential space. In *SWAT*, pages 125–129. IEEE, 1972. doi:10.1109/SWAT.1972.29.

[Nak17] Yoshiki Nakamura. Partial derivatives on graphs for Kleene allegories. In *LICS*, pages 1–12. IEEE, 2017. doi:10.1109/LICS.2017.8005132.

[O'H20] Peter W. O'Hearn. Incorrectness logic. *Proc. ACM Program. Lang.*, 4(POPL):10:1–10:32, 2020. doi:10.1145/3371078.

[Pou13] Damien Pous. Kleene Algebra with Tests and Coq tools for while programs. In *ITP*, volume 7998 of *LNCS*, pages 180–196. Springer, 2013. doi:10.1007/978-3-642-39634-2_15.

[Pou18] Damien Pous. On the positive calculus of relations with transitive closure. In *STACS*, volume 96 of *LIPIcs*, pages 3:1–3:16. Schloss Dagstuhl, 2018. doi:10.4230/LIPIcs.STACS.2018.3.

[Pra80] Vaughan R. Pratt. Dynamic algebras and the nature of induction. In *ACM Symposium on Theory of Computing*, STOC '80, page 22–28, New York, NY, USA, 1980. Association for Computing Machinery. doi:10.1145/800141.804649.

[PRW21] Damien Pous, Jurriaan Rot, and Jana Wagemaker. On tools for completeness of Kleene algebra with hypotheses. In *RAMiCS*, volume 13027 of *LNCS*, pages 378–395. Springer, 2021. doi:10.1007/978-3-030-88701-8_23.

[PRW24] Damien Pous, Jurriaan Rot, and Jana Wagemaker. On tools for completeness of kleene algebra with hypotheses. *Logical Methods in Computer Science*, 20(2), 2024. doi:10.46298/LMCS-20(2:8)2024.

[PW22] Damien Pous and Jana Wagemaker. Completeness theorems for Kleene algebra with top. In *CONCUR 2022*, volume 243 of *LIPIcs*. Schloss Dagstuhl, 2022. doi:10.4230/LIPIcs.CONCUR.2022.26.

[Red64] V.N. Redko. On the algebra of commutative events. *Ukrainian Math. Zh.*, 16:185–195, 1964.

[Sak09] Jacques Sakarovitch. *Elements of Automata Theory*. Cambridge University Press, 2009. doi:10.1017/CBO9781139195218.

[Sav70] 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.

[Tho68] Ken Thompson. Regular expression search algorithm. *Communications of the ACM*, 11:419–422, 1968. URL: http://www.fing.edu.uy/inco/cursos/intropln/material/p419-thompson.pdf.

[ZdAG22] Cheng Zhang, Arthur Azevedo de Amorim, and Marco Gaboardi. On incorrectness logic and Kleene algebra with top and tests. *Proc. ACM Program. Lang.*, 6(POPL):1–30, 2022. doi:10.1145/3498690.

Appendix A. Proof of Theorem 4.5

We give here a proof of Theorem 4.5. Variants of this theorem appeared for Kleene allegories without top in [BP15, Theorem 6], and for Kleene allegories with top in [Pou18, Theorem 16].

First we observe that valuations into relational models are very close to (potentially infinite) graphs in the sense of Definition 4.3: it suffices to adjoin to them an input and an output.

Definition A.1 (Graph of a valuation)**.** Consider a member of REL: relations on some set X with a function $p\colon X \to \mathsf{At}$. Let $\sigma\colon \Sigma \to \mathcal{P}(X\times X)$ be a valuation of Σ into relations on X. For all elements $i, j \in X$, we define the graph $\langle \sigma, i, j\rangle \triangleq \langle X, F, p, i, j\rangle$ where $F \triangleq \{\langle x, a, y\rangle \mid a \in \Sigma,\ \langle x, y\rangle \in \sigma(a)\}$.

The first key lemma characterises evaluation of expressions not using $0, +, \cdot^*$ in a relational model, in terms of graph homomorphisms. In our case, expressions not using $0, +, \cdot^*$ can be represented by guarded strings. Such a lemma appeared first in [AB95, Lemma 3] for a signature including intersection and converse, but not top nor tests. Under its original formulation, its extension to cover top is trivial once we realise that the graph of $\top$ should simply be a graph without edges and exactly two vertices (the input and the output).

Lemma A.2. *Let $\sigma\colon \Sigma \to \mathcal{P}(X\times X)$ be a valuation of Σ into a member of REL. For all guarded strings u, we have*

$$\langle i, j\rangle \in \hat{\sigma}(u) \iff \langle \sigma, i, j\rangle \lhd g(u)$$

Proof. By induction on u.

- if $u = \alpha$ is an atom, then both sides reduce to the condition $i = j \wedge p(i) = \alpha$;
- if $u = \alpha a \beta$ has length one
  - if $a = \top$, then both sides reduce to the condition $p(i) = \alpha \wedge p(j) = \beta$;
  - if $a \in \Sigma$, then both sides reduce to the conjunction of the previous condition and $\langle i, j\rangle \in \sigma(a)$;
- if $u = v\alpha w$ for two smaller guarded strings $v\alpha$ and αw then we have

$$\begin{aligned}
&\langle i, j\rangle \in \hat{\sigma}(v\alpha w)\\
\Leftrightarrow\ &\exists k,\ \langle i, k\rangle \in \hat{\sigma}(v)\ \wedge\ p(k) = \alpha\ \wedge\ \langle k, j\rangle \in \hat{\sigma}(\alpha w) && \text{(by definition)}\\
\Leftrightarrow\ &\exists k,\ \langle i, k\rangle \in \hat{\sigma}(v\alpha)\ \wedge\ \langle k, j\rangle \in \hat{\sigma}(\alpha w)\\
\Leftrightarrow\ &\exists k,\ \langle \sigma, i, k\rangle \lhd g(v\alpha)\ \wedge\ \langle \sigma, k, j\rangle \lhd g(\alpha w) && \text{(by induction hypothesis on } v\alpha \text{ and } \alpha w)\\
\Leftrightarrow\ &\langle \sigma, i, j\rangle \lhd g(v\alpha w)
\end{aligned}$$

(The last equivalence comes from a simple analysis of the homomorphisms whose source is a sequential composition of two graphs—see, e.g., [AB95, Lemma 2(ii)].) □

The second key lemma characterises the evaluation of an arbitrary expression in terms of (the evaluations of) the guarded strings in the language of that expression. Variants of such a lemma often appear in the literature for *star-continuous* models, rather than just relational ones (e.g., [KS96, Lemma 4]).

Lemma A.3. *Let $\sigma\colon \Sigma \to \mathcal{P}(X\times X)$ be a valuation of Σ into a member of* REL*. For all expressions e, we have*

$$\hat{\sigma}(e) = \bigcup_{u\in[e]} \hat{\sigma}(u)$$

Proof. By an easy induction on e, using distributivity of $\cdot$ over arbitrary unions in REL. □

Equipped with those two lemmas, we obtain the announced theorem.

Theorem A.4. *For all expressions e, f, we have:*

$$\mathsf{REL} \models e \leq f \iff \forall u \in [e],\ \exists v \in [f],\ g(u) \lhd g(v)$$

Proof. For the forward implication, assume $\mathsf{REL} \models e \leq f$ and let $u \in [e]$. Let n be the length of u and consider relations on $[0; n]$, a member of REL with the function p mapping $i \leq n$ to the ith atom in u. Define $\sigma\colon \Sigma \to \mathcal{P}([0; n]\times[0; n])$ by $\langle i, j\rangle \in \sigma(a)$ if the i-th letter of u is a and $j = i + 1$. The graph $g(u)$ is nothing but $\langle \sigma, 0, n\rangle$, so that we have $\langle 0, n\rangle \in \hat{\sigma}(u)$ by Lemma A.2, using the identity graph homomorphism. Thus we consecutively get $\langle 0, n\rangle \in \hat{\sigma}(e)$ by Lemma A.3, $\langle 0, n\rangle \in \hat{\sigma}(f)$ by assumption, and $\langle 0, n\rangle \in \hat{\sigma}(v)$ for some $v \in [f]$ by Lemma A.3 again. Lemma A.2 finally gives $g(u) = \langle \sigma, 0, n\rangle \lhd g(v)$, as required.

For the backward implication, assume the right-hand side and let $\sigma\colon \Sigma \to \mathcal{P}(X\times X)$ be a valuation into a member of REL. For all $i, j \in X$, we have

$$\begin{aligned}
&\langle i, j\rangle \in \hat{\sigma}(e) \\
\Leftrightarrow\ &\langle i, j\rangle \in \hat{\sigma}(u) \text{ for some } u \in [e] && \text{(by Lemma A.3)} \\
\Leftrightarrow\ &\langle \sigma, i, j\rangle \lhd g(u) \text{ for some } u \in [e] && \text{(by Lemma A.2)} \\
\Rightarrow\ &\langle \sigma, i, j\rangle \lhd g(u) \text{ for some } u, v \text{ s.t. } v \in [f] \text{ and } g(u) \lhd g(v) && \text{(by assumption)} \\
\Rightarrow\ &\langle \sigma, i, j\rangle \lhd g(v) \text{ for some } v \in [f] && \text{(by transitivity of } \lhd) \\
\Leftrightarrow\ &\langle i, j\rangle \in \hat{\sigma}(v) \text{ for some } v \in [f] && \text{(by Lemma A.2)} \\
\Leftrightarrow\ &\langle i, j\rangle \in \hat{\sigma}(f) && \text{(by Lemma A.3)}
\end{aligned}$$

Whence $\hat{\sigma}(e) \subseteq \hat{\sigma}(f)$, and thus $\mathsf{REL} \models e \leq f$ as required. □