The Equational Theory of Relational Kleene Algebra with Graph Loop is PSPACE-Complete

Yoshiki Nakamura
Chiba University, Japan

Abstract

In this paper, we show that the equational theory of relational Kleene algebra with the *graph loop* operator (a.k.a. *fixset*) is PSPACE-complete. Here, the graph loop is the unary operator that restricts a binary relation to the identity relation. We further show that this PSPACE-completeness still holds by extending the terms with top, tests, converse, and nominals, over relational models. Notably, for Kleene algebra with tests (KAT), while the equational theory of relational KAT with antidomain is EXPTIME-complete, we show that the equational theory of relational KAT with domain is PSPACE-complete, thereby resolving a problem left open in previous works.

To this end, we introduce a novel automaton model on relational structures (graphs), called *loop-automata*. Loop-automata extend nondeterministic finite automata with a transition type that tests whether the current vertex has a loop. Using this model, we can give a polynomial-time reduction from the equational theories above to the language inclusion problem for 2-way alternating automata.

2012 ACM Subject Classification Theory of computation → Logic; Theory of computation → Formal languages and automata theory; Theory of computation → Logic and databases; Theory of computation → Programming logic

Keywords and phrases Kleene algebra, Graph loop, Domain

Digital Object Identifier 10.4230/LIPIcs.FSCD.2026.25

Related Version *Full Version*: `https://arxiv.org/abs/2512.22930`

Acknowledgements This work was supported by JSPS KAKENHI Grant Numbers JP21K13828 and JP25K14985.

1 Introduction

Relational Kleene algebra (*RKA*), *Kleene algebra* over relational models, is an algebraic system on binary relations with the operators $\{0, 1, ;, +, \cdot^*\}$ of empty relation (0), identity relation (1), union ($+$), composition ($;$), and reflexive transitive closure (Kleene star) ($\cdot^*$). RKA arises in various research areas, for instance, in database theory via *regular path queries* (RPQs) [5] and in program verification via *relational Kleene algebra with tests* (*relational KAT*, *RKAT*) [19]. The equational theory of RKA is well-known to be decidable and PSPACE-complete, because it coincides[1] with the language equivalence problem of regular expressions (which is PSPACE-complete [23]). The equational theory over relational models is still PSPACE-complete even if we extend RKA terms with several operators, such as tests [19], converse [3], top [25, 31], nominals [27], and all the combinations of them [27, Cor. 6.7].

A remarkable extension is RKA with *domain* ($\cdot^{\mathsf{d}}$) and *antidomain* ($\cdot^{\mathsf{a}}$) operators [8, 9, 10]. The domain operator $\cdot^{\mathsf{d}}$ maps a binary relation R (on a set X) to the diagonal of the domain and the antidomain operator $\cdot^{\mathsf{a}}$ maps R to the (unary) complement of R^{d}:

$$R^{\mathsf{d}} \coloneqq \{(x,x) \mid x \in X,\ \exists y \in X.(x,y) \in R\}, \quad R^{\mathsf{a}} \coloneqq \{(x,x) \mid x \in X,\ \forall y \in X.(x,y) \notin R\}.$$

[1] This coincidence can be seen, *e.g.*, from the completeness theorem of Kleene algebra [17] or from a straightforward argument regarding labelled paths in (relational) structures.

The equational theory of RKA with antidomain[2] is ExpTime-complete [35], coinciding with the complexity of the theory of Propositional Dynamic Logic (PDL). However, for the equational theory of RKA with domain, the complexity was left open by McLean [21, Problem 7.3] and Sedlár [34, p. 16]. To the best of our knowledge,[3] it was previously known only that the problem is PSpace-hard [23] and in ExpTime [35].

Another extension is RKA with *intersection* ($\cap$), a.k.a. (representable) *relational Kleene lattice* [1]. The equational theory is ExpSpace-complete [24, 4, 27], while it is PSpace-complete if we fix the intersection width (*i.e.*, the maximum nesting depth of $\cap$) of input terms [27]. RKA with *graph loop* ($\cdot^{\circlearrowleft}$) –henceforth, *loop-RKA* for short– [27] is a restriction of RKA with intersection, where we only allow the intersection with identity relation (*i.e.*, the graph loop operator). Here, graph loop[4] [7, 26] (also called *fixset* [15]) is the unary operator of restricting a given binary relation R on X to the identity relation: $R^{\circlearrowleft} \coloneqq R \cap \triangle_X$. Loop-RKA is still expressive enough to define domain and range operators using top ($\top$), the full relation, as follows:

$$t^{\mathsf{d}} \;=\; (t\,;\top)^{\circlearrowleft}, \qquad\qquad t^{\mathsf{r}} \;=\; (\top\,;t)^{\circlearrowleft}. \qquad\qquad (\mathsf{d}/\mathsf{r}\text{-by-}\circlearrowleft,\top)$$

However, the complexity of the equational theory of loop-RKA (without $\top$) was also left open by Nakamura [27, §8]. To the best of our knowledge, it was previously known only that the problem is PSpace-hard [23] and in ExpTime [7, 13]. *Loop-PDL* [7] is propositional dynamic logic of regular programs with graph loop (strong loop predicate). The theory of loop-PDL is ExpTime-complete [7] (another proof is in [13, Corollary 4.9]). By a canonical embedding (*cf.*, [11, p. 209]), we can give a polynomial-time reduction from the equational theory of loop-RKA into the theory of loop-PDL. The ExpTime upper bound for loop-RKA can be obtained via this reduction. For loop-PDL, the ExpTime upper bound can also be obtained from the theory of PDL with intersection where the intersection width is fixed [13, Corollary 4.9]. In contrast, we cannot obtain the PSpace upper bound of loop-RKA from the equational theory of RKA with intersection where the intersection width is fixed, because "rich tests" in the context of PDL are not employed in RKA with intersection.[5]

Contributions. We first show the following, which affirmatively answers [27, §8]:

▶ **Theorem 1.** *The equational theory for loop-RKA is* PSpace*-complete.*

Moreover, by adapting the encodings of some operators from [27, §6] in our automata construction, the PSpace upper bound is strengthened as follows (§ 7). Here, tests are in the context of KAT [19] and nominals are in the context of hybrid modal logic [2].

---

[2] The domain $\cdot^{\mathsf{d}}$ can be expressed using antidomain as $R^{\mathsf{d}} = (R^{\mathsf{a}})^{\mathsf{a}}$. The converse does not hold in general. Using complement $R^{-} \coloneqq X^2 \setminus R$ where X is the base set, the antidomain $\cdot^{\mathsf{a}}$ can be expressed as $R^{\mathsf{a}} = ((R^{\mathsf{d}})^{-})^{\mathsf{d}}$. However, the equational theory of RKA with complement is undecidable (by [14]). It is still Π^0_1-complete even when the application of complement is restricted to constants [28, Corollary 7.2] or variables [25, Theorem 50] (via the reduction of [28, Example 3.4] from Horn formulas to equations).

[3] In [33, unpublished manuscript] it is claimed that the containment problem of *nested regular expressions* [32] (*i.e.*, the inequational theory of RKA with domain and converse) is in PSpace, which also implies the PSpace upper bound of the equational theory of RKA with domain. However, in the version of [33], the containment on "semipaths" (not general structures) is only considered as noted in [33, p. 1].

[4] In [7], the *strong loop predicate* loop(t) is considered. In our paper, we use it as operator. Precisely, $t^{\circlearrowleft}$ is equal to loop(t)? where $\varphi?$ denotes the *rich test*.

[5] For loop-PDL, we can reduce the intersection width to at most 2 [13, Corollary 4.9], *e.g.*, by translating the nested $(ab^{\circlearrowleft}a)^{\circlearrowleft}$ into $(a(\langle b^{\circlearrowleft}\rangle\mathsf{true})?a)^{\circlearrowleft}$, where $\varphi?$ denotes the rich test and $\langle t\rangle\varphi$ denotes the diamond modality. Note that the intersection width of $\varphi?$ is always defined to be 1, even if φ contains arbitrary nested graph loop.

▶ **Theorem 2.** *The equational theory for loop-RKA with top, tests, converse, and nominals is* PSPACE*-complete.*

As an application, we can also encode domain and range as above (d/r-by-$\circlearrowleft$,$\top$). Consequently, our result also implies the PSPACE upper bounds for RKAT with domain and RKA with domain [10, 21, 34][6], which answers [21, Problem 7.3] and [34, p. 16]:

▶ **Corollary 3.** *The equational theory for RKA with tests, domain, and range is* PSPACE*-complete.*

Notably, adding domain to RKA maintains the PSPACE upper bound, while the equational theory of RKA with antidomain ($\cdot^{\mathsf{a}}$) is EXPTIME-complete [35].

To prove Theorems 1 and 2, we give a reduction from the equational theory into the language inclusion problem of 2-way alternating string automata (2AFAs) [12]. We first observe that the equational theory has the *linearly bounded pathwidth model property* (Proposition 7) [27, Proposition 2.9]: if there is some counter-model for an equation (*i.e.*, a structure refuting the equation), then there is also some counter-model with pathwidth linear in the size of the equation. By viewing path decompositions of bounded width as strings, we can enumerate all possible structures of bounded pathwidth, using string automata. This baseline approach itself is the same as in [27] for RKA with intersection. However, the construction of [27] does not imply the PSPACE upper bound (see also [27, p. 46–47]), especially if the intersection width is not fixed.

To obtain a PSPACE-algorithm, we introduce a novel automaton model, called *loop-automata* (§ 4) as an alternative expression of the relational semantics of loop-RKA. Loop-automata are obtained from nondeterministic automata by adding a new transition for testing whether there exists a loop in the current vertex. From loop-automata, we build 2AFAs, where each string expresses a path decomposition. Using this construction, we give a reduction from the equational theory of loop-RKA into the language inclusion problem of 2AFAs. In our automata construction, we can also encode several additional operators, which shows Theorem 2 and Corollary 3.

Related and future work. Our automata construction in this paper is based on [27], which shows the equational theory for the *positive* fragment of Tarski's calculus of relations with transitive closure (*PCoR**) [36, 30, 27] is EXPSPACE-complete. While loop-RKA can be viewed as a syntactic fragment of PCoR*, the algorithm presented in [27] requires an exponential space even for loop-RKA [27, p. 46–47]. Our automata construction in this paper refines it, specializing loop-RKA. Our reduction is different from the reduction of [27], mainly in the following two points:

- We consider only (nested) path graphs instead of *series-parallel graphs*, using loop-automata instead of *derivatives* on graphs. These graphs are sufficient for loop-RKA. This significantly simplifies our algorithm compared to that of [27] and enables us to obtain the PSPACE upper bound.
- We reduce the alphabet size using a binary encoding of structures. This is also crucial for our PSPACE upper bound, especially if the intersection width is not fixed.

Notably, RKAT with domain and range has a connection with *incorrectness logic* [29]. An (angelic total) incorrectness triple (*i.e.*, all states in q can be reached by running t on some state in p) can be expressed as follows, where p, q are tests and t is a KAT term:

$$[p]\ t\ [q] \quad \leftrightarrow \quad (ptq)^{\mathsf{r}} \geq q \quad (\leftrightarrow \quad (ptq)^{\mathsf{r}} = q^{\mathsf{r}} \quad \leftrightarrow \quad \top ptq = \top q).$$

[6] RKA with domain is a strictly less expressive fragment of RKA with tests and antidomain. The latter system is often also called (relational) "Kleene algebra with domain" [8].

See also [38, Table 2] for further correspondences between Hoare-like logics and RKAT with domain and range, based on the equivalences $t\top = s\top \leftrightarrow t^{\mathsf{d}} = s^{\mathsf{d}}$ and $\top t = \top s \leftrightarrow t^{\mathsf{r}} = s^{\mathsf{r}}$. In previous work, the PSPACE upper bound of the validity of such incorrectness triples was shown via an encoding to KAT with top over relational models [25, 31], or KAT with top over language models [39]. Our Corollary 3 more directly shows that the above (in)equation is in PSPACE.

In this paper, we present an algorithm for loop-RKA; however, it would also be interesting to present an algorithm specialized to RKA with domain.[7]

Additionally, it is open whether the equational theory of loop-RKA (*resp.*, *loop-RKAT*, *i.e.*, RKAT with graph loop) is finitely axiomatizable. An interesting valid equation in loop-RKA is $(t^+)^{\circlearrowleft} \leq (tt)^+$ (cf. [30, p. 14]).

Organization. In § 2, we give basic definitions. In § 3, we define loop-RKA. In § 4, we introduce loop-automata as an alternative of loop-RKA terms. In § 5 and 6, we show that the equational theory of loop-RKA is PSPACE-complete (Theorem 1). In § 7, we extend the automata construction with several additional operators (Theorem 2 and Corollary 3).

2 Preliminaries

We write $\mathbb{N}$ for the set of non-negative integers. For $l, r \in \mathbb{N}$, we write $[l\,..\,r]$ for the set $\{i \in \mathbb{N} \mid l \leq i \leq r\}$. For $n \in \mathbb{N}$, we abbreviate $[1\,..\,n]$ to $[n]$. For a set X, we write $\#X$ for the cardinality of X. We may use $\sqcup$ to denote that the union $\cup$ is disjoint, explicitly. For a sequence $\bar{a} = a_1 \dots a_n$, we write $\bar{a}[i]$ for the element a_i.

2.1 Word languages

For a set X of *characters*, we write X^* for the set of *strings* over X. We write wv for the *concatenation* of strings w and v. We write ε for the *empty string*. A *language* over X is a subset of X^*. We use w, v to denote strings and use $\mathcal{L}, \mathcal{K}$ to denote languages. For languages $\mathcal{L}, \mathcal{K} \subseteq X^*$, the *concatenation* $\mathcal{L}\,;\mathcal{K}$, the *n-th iteration* $\mathcal{L}^n$ (where $n \in \mathbb{N}$), the *Kleene plus* $\mathcal{L}^+$, and the *Kleene star* $\mathcal{L}^*$ are defined by:

$$\mathcal{L}\,;\mathcal{K} \coloneqq \{wv \mid w \in \mathcal{L} \ \wedge\ v \in \mathcal{K}\},$$

$$\mathcal{L}^n \coloneqq \begin{cases} \mathcal{L}\,;\mathcal{L}^{n-1} & (n \geq 1) \\ \{\varepsilon\} & (n = 0), \end{cases} \qquad \mathcal{L}^+ \coloneqq \bigcup_{n \geq 1} \mathcal{L}^n, \qquad \mathcal{L}^* \coloneqq \bigcup_{n \geq 0} \mathcal{L}^n.$$

2.2 Binary relations

We write $\triangle_A$ for the *identity relation* on a set A: $\triangle_A \coloneqq \{(x, x) \mid x \in A\}$. For *binary relations* R, S on a set B, the *composition* $R\,;S$, the *n-th iteration* R^n (where $n \in \mathbb{N}$), the *transitive closure* R^+, the *reflexive transitive closure* R^*, the *domain* R^{d}, the *range* R^{r}, and

[7] Our approach of encoding domain using graph loop and top does not preserve the complexity in general. For example, whereas the equational theory of RKA with domain and variable complements is in EXPTIME (by [20]), it is undecidable for loop-RKA with variable complements [25, 28] (Footnote 2).

the *graph loop* $R^{\circlearrowleft}$ are defined by:

$$R\,;S \coloneqq \{(x,z) \mid \exists y, (x,y) \in R \ \wedge\ (y,z) \in S\}, \qquad R^{\circlearrowleft} \coloneqq R \cap \triangle_B,$$

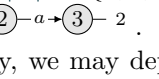
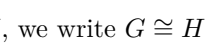
$$R^n \coloneqq \begin{cases} R\,;R^{n-1} & (n \geq 1) \\ \triangle_B & (n = 0) \end{cases}, \qquad R^+ \coloneqq \bigcup_{n\geq 1} R^n, \quad R^* \coloneqq \bigcup_{n\geq 0} R^n,$$

$$R^{\mathsf{d}} \coloneqq \{(x,x) \mid \exists y, (x,y) \in R\}, \qquad R^{\mathsf{r}} \coloneqq \{(y,y) \mid \exists x, (x,y) \in R\}.$$

2.3 Graphs and structures

For $k \geq 0$ and a set A, a *graph* G over A with k *ports* (*i.e.*, k named vertices) is a tuple $(|G|, \{a^G\}_{a\in A}, 1^G, \ldots, k^G)$, where

- $|G|$ is a set (which is possibly empty when $k = 0$) of *vertices*,
- $a^G \subseteq |G|^2$ is a binary relation for each $a \in A$ denoting the set of *edges* labelled by a,
- $i^G \in |G|$ is the i-th port for each $i \in [k]$.

We depict graphs as usual. For instance, when G is defined as $|G| = \{1,2,3\}$, $a^G = \{(1,2),(2,3)\}$, $1^G = 1$, and $2^G = 3$, we depict G as  . By viewing the ports 1^G and 2^G as the "input" and "output", respectively, we may depict G as (by drawing  as  and as , respectively, and by forgetting vertex labels),  (by forgetting edge labels a), or  . For two graphs G and H, we write $G \cong H$ if they are graph isomorphic.

A *structure* over A is a non-empty graph without ports over A. We use $\mathcal{M}, \mathcal{N}$ to denote structures. We write REL for the class of all structures.

2.4 Pathwidth, path decomposition, and gluing operator

We recall the pathwidth of structures [6, Def. 9.12]. For a finite structure $\mathcal{M}$ over A, a *path decomposition* of $\mathcal{M}$ is a sequence $\bar{\mathcal{N}} = \mathcal{N}_1 \ldots \mathcal{N}_n$ of finite structures such that

- $|\mathcal{M}| = \bigcup_{i\in[n]} |\mathcal{N}_i|$ and $a^{\mathcal{M}} = \bigcup_{i\in[n]} a^{\mathcal{N}_i}$ for each $a \in A$,
- $|\mathcal{N}_i| \cap |\mathcal{N}_k| \subseteq |\mathcal{N}_j|$ for all $1 \leq i \leq j \leq k \leq n$.

Each $\mathcal{N}_i$ is called a *bag*. The *width* of $\bar{\mathcal{N}}$ is $\max_{i\in[n]}(\#|\mathcal{N}_i| - 1)$. The *pathwidth* $\mathrm{pw}(\mathcal{M})$ of a finite structure $\mathcal{M}$ is defined as the minimum width among path decompositions of $\mathcal{M}$. For instance, for the $\mathcal{M}$ and $\bar{\mathcal{N}}$ in Fig. 1, $\bar{\mathcal{N}}$ is a path decomposition of $\mathcal{M}$ of width 2.

In this paper, we will view path decompositions as *strings*, based on, *e.g.*, [24, 27]. We define $\odot\bar{\mathcal{M}}$ [27, Definition 5.1] as the structure obtained from the disjoint union of structures in $\bar{\mathcal{M}}$ by gluing vertices having the same name in adjacent structures. We write $[(i,x)]_\sim$ for the equivalence class of the vertex named by x in $\bar{\mathcal{M}}[i]$ *w.r.t.* the equivalence relation induced from the construction of $\odot\bar{\mathcal{M}}$; more precisely, it is the minimal equivalence relation $\sim$ satisfying $(i,v) \sim (i+1,v)$ for all $i \in [1\,..\,n-1]$ and $v \in |\bar{\mathcal{M}}[i]| \cap |\bar{\mathcal{M}}[i+1]|$.

▶ **Example 4.** For the $\bar{\mathcal{N}}$ and $\mathcal{M}$ in Fig. 1, the structure $\odot\bar{\mathcal{N}}$ is illustrated as in Fig. 1 by gluing vertices according to the dotted lines. We observe that the structure is isomorphic to $\mathcal{M}$. In general, the gluing operator transforms a given path decomposition into the original structure up to isomorphisms.

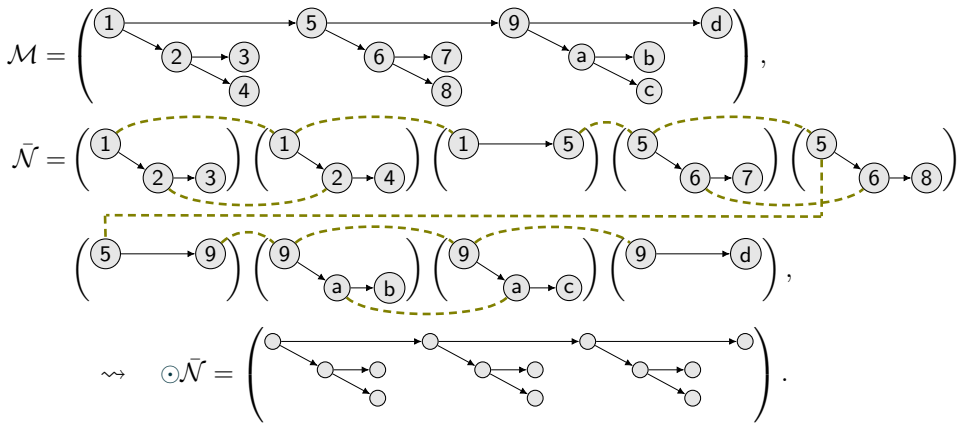
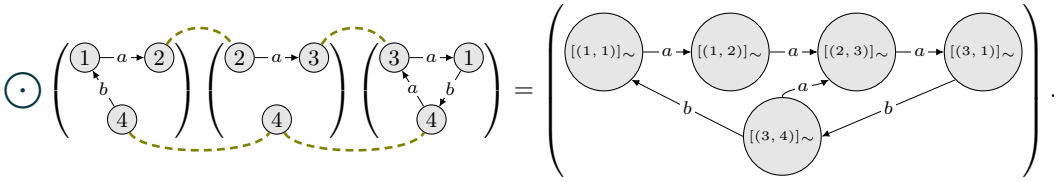


Figure 1 Illustrative example of path decomposition and the gluing operator $\odot$. $\bar{\mathcal{N}}$ is a path decomposition of $\mathcal{M}$. $\odot\bar{\mathcal{N}}$ is obtained from $\bar{\mathcal{N}}$ by gluing adjacent vertices with the same label (indicated by dotted lines).

▶ **Example 5.** Below is another example of the gluing operator $\odot$:



Note that the vertices labelled with 1 are not glued since they are not adjacent.

3 Relational Kleene Algebra with Graph Loop

In this section, we define *relational Kleene algebra with graph loop* (*loop-RKA*).

Syntax. Given a set Σ of *variables*, the *terms* over Σ are defined as the terms over the signature $\{0_{(0)}, 1_{(0)}, +_{(2)}, ;_{(2)}, \cdot^*_{(1)}, \cdot^{\circlearrowleft}_{(1)}\}$:

$$t, s, u \quad ::= \quad a \mid 0 \mid 1 \mid t + s \mid t\,;\,s \mid t^* \mid t^{\circlearrowleft}. \qquad \text{where } a \in \Sigma$$

We often abbreviate $t\,;\,s$ to ts. We use parentheses in ambiguous situations. We write $\sum_{i=1}^{n} t_i$ for the term $0 + t_1 + \cdots + t_n$. We also let $t^+ := tt^*$. An *equation* $t = s$ is a pair of terms. An *inequation* $t \leq s$ abbreviates the equation $t + s = s$. The *size* $\|t\|$ of a term t is defined as the number of symbols occurring in t.

Semantics. For a structure $\mathcal{M}$ and a term t, the *semantics*[8] $[\![t]\!]^{\mathcal{M}} \subseteq |\mathcal{M}|^2$ is defined as follows:

$$[\![a]\!]^{\mathcal{M}} := a^{\mathcal{M}}, \qquad [\![0]\!]^{\mathcal{M}} := \emptyset, \qquad [\![1]\!]^{\mathcal{M}} := \triangle_{|\mathcal{M}|}, \qquad [\![t + s]\!]^{\mathcal{M}} := [\![t]\!]^{\mathcal{M}} \cup [\![s]\!]^{\mathcal{M}},$$
$$[\![t\,;\,s]\!]^{\mathcal{M}} := [\![t]\!]^{\mathcal{M}}\,;\,[\![s]\!]^{\mathcal{M}}, \qquad [\![t^*]\!]^{\mathcal{M}} := ([\![t]\!]^{\mathcal{M}})^*, \qquad [\![t^{\circlearrowleft}]\!]^{\mathcal{M}} := [\![t]\!]^{\mathcal{M}} \cap \triangle_{|\mathcal{M}|}.$$

Recall that REL is the class of all structures. For $\mathcal{C} \subseteq \mathsf{REL}$, we write $\mathcal{C} \models t = s$ if $[\![t]\!]^{\mathcal{M}} = [\![s]\!]^{\mathcal{M}}$ for all $\mathcal{M} \in \mathcal{C}$. In the sequel, we mainly consider the *equational theory w.r.t.* REL.

[8] For notational convenience, we use structures as an alternative to algebras of binary relations.

Linearly bounded pathwidth model property. The *intersection width* $\mathrm{iw}(t)$ [13] (where the term $t^{\circlearrowleft}$ is viewed as $t \cap \mathsf{I}$) is defined as follows:

$$\mathrm{iw}(a) \coloneqq 1 \text{ for } a \in \Sigma \cup \{\mathsf{I}, 0\}, \qquad \mathrm{iw}(t^*) \coloneqq \mathrm{iw}(t),$$
$$\mathrm{iw}(t \heartsuit s) \coloneqq \max(\mathrm{iw}(t), \mathrm{iw}(s)) \text{ for } \heartsuit \in \{;, +\}, \qquad \mathrm{iw}(t^{\circlearrowleft}) \coloneqq \mathrm{iw}(t) + 1.$$

For instance, $\mathrm{iw}(a^{\circlearrowleft}b^{\circlearrowleft}c^{\circlearrowleft}) = 2$ and $\mathrm{iw}((c(ba^{\circlearrowleft}b)^{\circlearrowleft}c)^{\circlearrowleft}) = 4$.

▶ **Proposition 6.** *For all terms t, we have $\mathrm{iw}(t) \leq \|t\|$.*

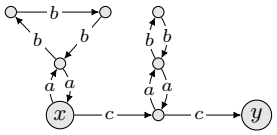
Proof. By easy induction on t. ◀

For a class $\mathcal{C} \subseteq \mathsf{REL}$, we let $\mathcal{C}_{\mathrm{pw}\leq k} = \{\mathcal{M} \in \mathcal{C} \mid \mathrm{pw}(\mathcal{M}) \leq k\}$. Because loop-RKA can be viewed as a syntactic fragment of PCoR* [30, 27], we have the *linearly bounded pathwidth model property*:

▶ **Proposition 7** ([27, Proposition 2.9]). *For all loop-RKA terms, t and s, we have:*

$$\mathsf{REL} \models t \leq s \iff \mathsf{REL}_{\mathrm{pw}\leq \mathrm{iw}(t)} \models t \leq s.$$

For instance, when $t = ((a(b^+)^{\circlearrowleft}a)^{\circlearrowleft}c)^+$ and $s = (a(b{+}bb)ac^*){+}(cabac^*)$, to show $\mathsf{REL} \not\models t \leq s$, it suffices to consider structures in $\mathsf{REL}_{\mathrm{pw}\leq 3}$ by $\mathrm{iw}(t) = 3$. For instance, the following structure $\mathcal{M} \in \mathsf{REL}_{\mathrm{pw}\leq 3}$ satisfies $(x, y) \in [\![t]\!]^{\mathcal{M}} \setminus [\![s]\!]^{\mathcal{M}}$ (hence, $\mathsf{REL} \not\models t \leq s$):

$\mathcal{M} =$ 

4 Automata Expression for Loop-RKA

In this section, we introduce an automata model, named *loop-automata*, for representing the relational semantics of loop-RKA. Loop-automata accept/reject ordered pairs of vertices in graphs (structures). Given two finite sets Σ (of characters) and Q (of *states*), we consider the following set of transition labels:

$$L_{\Sigma,Q} \coloneqq \Sigma \sqcup \{\mathsf{I}\} \sqcup \{\ell_{(p,q)} \mid (p, q) \in Q^2\}.$$

Each label $a \in \Sigma$ is employed for the standard transitions. The label I is employed for the "epsilon"-transitions: transitions that do not consume any input symbol. Each label $\ell_{(p,q)}$ is employed for expressing *loop-transitions*, which are conditional "epsilon"-transitions: the transition is allowed only when there exists a run from the state p on the current vertex to q on the same vertex.

▶ **Definition 8.** *Let Σ be a finite set of characters. A (non-deterministic)* loop-automaton *$\mathcal{A}$ over Σ is a graph over $L_{\Sigma,|\mathcal{A}|}$ with 2 ports, i.e., a tuple $(|\mathcal{A}|, \{\alpha^{\mathcal{A}}\}_{\alpha \in L_{\Sigma,|\mathcal{A}|}}, 1^{\mathcal{A}}, 2^{\mathcal{A}})$ where*

- *$|\mathcal{A}|$ is a finite set of states,*
- *$\alpha^{\mathcal{A}} \subseteq |\mathcal{A}|^2$ is a binary relation for expressing a transition function for each $\alpha \in L_{\Sigma,|\mathcal{A}|}$,*
- *$1^{\mathcal{A}} \in |\mathcal{A}|$ is the source state,*
- *$2^{\mathcal{A}} \in |\mathcal{A}|$ is the target state.*

Given a loop-automaton $\mathcal{A}$ and a structure $\mathcal{M}$ over a finite set Σ, an *$\mathcal{M}$-run* (or, run if $\mathcal{M}$ is clear from the context) τ of $\mathcal{A}$ is a non-empty sequence $x_0\alpha_1x_1 \dots x_{n-1}\alpha_nx_n$, where $x_i \in |\mathcal{M}|$ and $\alpha_i \in L_{\Sigma,|\mathcal{A}|}$ for each i. We write $1^{\tau} = x_0$ and $2^{\tau} = x_n$. For two $\mathcal{M}$-runs

$\tau = x_0\alpha_1 \dots \alpha_n x_n$ and $\sigma = y_0\beta_1 \dots \beta_m y_m$, the *coalesced product* $\tau \diamond \sigma$ is partially defined as follows: $\tau \diamond \sigma := x_0\alpha_1 \dots \alpha_n x_n\beta_1 \dots \beta_m y_m$ if $x_n = y_0$ and undefined otherwise. The relation $p \longrightarrow_\tau^{\mathcal{M},\mathcal{A}} q$ (or, $p \longrightarrow_\tau^{\mathcal{M}} q$ if $\mathcal{A}$ is clear) is defined as the minimal ternary relation of $p, q \in |\mathcal{A}|$ and a run τ, closed under the following rules:

$$\frac{p \in |\mathcal{A}| \text{ and } x \in |\mathcal{M}|}{p \longrightarrow_x^{\mathcal{M}} p}\ \mathrm{R} \qquad \frac{(p,q) \in \alpha^{\mathcal{A}} \text{ and } (x,y) \in [\![\alpha]\!]^{\mathcal{M}}}{p \longrightarrow_{x\alpha y}^{\mathcal{M}} q}\ \alpha \text{ for } \alpha \in \Sigma \sqcup \{1\}$$

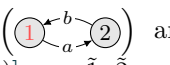
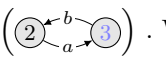
$$\frac{p \longrightarrow_\tau^{\mathcal{M}} q \quad q \longrightarrow_\sigma^{\mathcal{M}} r}{p \longrightarrow_{\tau\diamond\sigma}^{\mathcal{M}} r}\ \mathrm{T} \qquad \frac{(p,q) \in \ell^{\mathcal{A}}_{(p',q')},\ p' \longrightarrow_\tau^{\mathcal{M}} q', \text{ and } 1^\tau = 2^\tau = x \in |\mathcal{M}|}{p \longrightarrow_{x\ell_{(p',q')}x}^{\mathcal{M}} q}\ \ell_{(p',q')}$$

The semantics $[\![\mathcal{A}]\!]^{\mathcal{M}} \subseteq |\mathcal{M}|^2$ is given as follows:

$$[\![\mathcal{A}]\!]^{\mathcal{M}} := \{(x,y) \in |\mathcal{M}|^2 \mid 1^{\mathcal{A}} \longrightarrow_\tau^{\mathcal{M}} 2^{\mathcal{A}} \text{ for some run } \tau \text{ s.t. } 1^\tau = x \text{ and } 2^\tau = y\}.$$

▶ **Example 9.** Let $t := aa(ba)^{\circlearrowleft}bb$. Then the following $\mathcal{A}_t$ is the corresponding loop-automaton (i.e., $[\![\mathcal{A}_t]\!]^{\mathcal{M}} = [\![t]\!]^{\mathcal{M}}$ holds for every structure $\mathcal{M}$):

$$\mathcal{A}_t := \Bigg(\quad \begin{array}{ccccccccccccc} & & q_1 & & b & & q_2 & & a & & q_3 & & \\ p_1 & a & p_2 & a & p_3 & \ell_{(q_1,q_3)} & p_4 & b & p_5 & b & p_6 & & \end{array} \quad \Bigg).$$

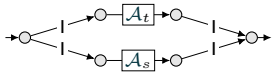
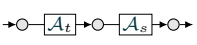
For instance, let $\mathcal{M} := \odot(\mathcal{N}_1\mathcal{N}_2)$ where $\mathcal{N}_1 =$  and $\mathcal{N}_2 =$  . We just write $[(1,1)]_\sim$, $[(1,2)]_\sim$ $(= [(2,2)]_\sim)$, and $[(2,3)]_\sim$, as $\tilde{1}$, $\tilde{2}$, and $\tilde{3}$, respectively, for short. We then have $(\tilde{1},\tilde{1}) \in [\![t]\!]^{\mathcal{M}}$. We also have $(\tilde{1},\tilde{1}) \in [\![\mathcal{A}_t]\!]^{\mathcal{M}}$, *e.g.*, by the following derivation tree:

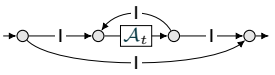
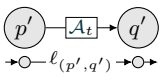
$$\dfrac{\dfrac{p_1 \longrightarrow_{\tilde{1}a\tilde{2}}^{\mathcal{M}} p_2 \quad p_2 \longrightarrow_{\tilde{2}a\tilde{3}}^{\mathcal{M}} p_3}{p_1 \longrightarrow_{\tilde{1}a\tilde{2}a\tilde{3}}^{\mathcal{M}} p_3}\ \mathrm{T} \qquad \dfrac{\dfrac{q_1 \longrightarrow_{\tilde{3}b\tilde{2}}^{\mathcal{M}} q_2 \quad q_2 \longrightarrow_{\tilde{2}a\tilde{3}}^{\mathcal{M}} q_3}{q_1 \longrightarrow_{\tilde{3}b\tilde{2}a\tilde{3}}^{\mathcal{M}} q_3}\ \mathrm{T}}{p_3 \longrightarrow_{\tilde{3}\ell_{(q_1,q_3)}\tilde{3}}^{\mathcal{M}} p_4}\ \ell_{(q_1,q_3)} \qquad \dfrac{p_4 \longrightarrow_{\tilde{3}b\tilde{2}}^{\mathcal{M}} p_5 \quad p_5 \longrightarrow_{\tilde{2}b\tilde{1}}^{\mathcal{M}} p_6}{p_4 \longrightarrow_{\tilde{3}b\tilde{2}b\tilde{1}}^{\mathcal{M}} p_6}\ \mathrm{T}}{p_1 \longrightarrow_{\tilde{1}a\tilde{2}a\tilde{3}\ell_{(q_1,q_3)}\tilde{3}b\tilde{2}b\tilde{1}}^{\mathcal{M}} p_6}\ \mathrm{T}}{}\ \mathrm{T}$$

Generalizing Example 9, we can transform every loop-RKA term into a loop-automaton, by extending the McNaughton–Yamada–Thompson construction [22, 37] for the graph loop $t^{\circlearrowleft}$.

▶ **Definition 10.** *For a loop-RKA term t, we define the loop-automaton $\mathcal{A}_t$, as follows:*[9]

$\mathcal{A}_a :=$, $\mathcal{A}_0 :=$, $\mathcal{A}_1 :=$,

$\mathcal{A}_{t+s} :=$  , $\mathcal{A}_{t;s} :=$  ,

$\mathcal{A}_{t^*} :=$  , $\mathcal{A}_{t^{\circlearrowleft}} :=$  where p', q' are fresh.

▶ **Proposition 11.** *For every term t and every structure $\mathcal{M}$, we have $[\![t]\!]^{\mathcal{M}} = [\![\mathcal{A}_t]\!]^{\mathcal{M}}$.*

Proof. By easy induction on t. As the other cases are the same as the McNaughton–Yamada–Thompson construction, we only write the case of $t = s^{\circlearrowleft}$. First, we have both $[\![s^{\circlearrowleft}]\!]^{\mathcal{M}} \subseteq \triangle_{|\mathcal{M}|}$ and $[\![\mathcal{A}_{s^{\circlearrowleft}}]\!]^{\mathcal{M}} \subseteq \triangle_{|\mathcal{M}|}$ by the form of $s^{\circlearrowleft}$ and $\mathcal{A}_{s^{\circlearrowleft}}$, respectively. We also

[9] For simplicity, this graphical definition is given modulo renaming of states by hiding names (note that $\ell_{p,q}$ also uses names of states).

have: $(x,x) \in [\![s^{\circlearrowleft}]\!]^{\mathcal{M}}$ *iff* $(x,x) \in [\![s]\!]^{\mathcal{M}}$ *iff* $(x,x) \in [\![\mathcal{A}_s]\!]^{\mathcal{M}}$ (IH) *iff* there is a run τ of $\mathcal{A}_s$ on $\mathcal{M}$ such that $1^\tau = 2^\tau = x$ *iff* the sequence $1^{\mathcal{A}_{s^{\circlearrowleft}}} \ell_{(1^{\mathcal{A}_s}, 2^{\mathcal{A}_s})} 2^{\mathcal{A}_{s^{\circlearrowleft}}}$ is a run of $\mathcal{A}_{s^{\circlearrowleft}}$ on $\mathcal{M}$ (by construction of $\mathcal{A}_{s^{\circlearrowleft}}$) *iff* $(x,x) \in [\![\mathcal{A}_{s^{\circlearrowleft}}]\!]^{\mathcal{M}}$. Hence, this case has been proved. ◀

From Proposition 11, in the sequel, we use loop-automata instead of terms in loop-RKA.

5 Decomposition Theorem

Let $\bar{\mathcal{M}} \in \mathsf{REL}^+$. We consider the glued structure $\odot\bar{\mathcal{M}}$ (recall § 2). For each $i \in [n]$, we define the 3-partition $\{|\bar{\mathcal{M}}|_{(i)}, |\bar{\mathcal{M}}|_{(i<)}, |\bar{\mathcal{M}}|_{(i>)}\}$ of $|\odot\bar{\mathcal{M}}|$:

$$|\bar{\mathcal{M}}|_{(i)} \coloneqq \{[(i,x)]_{\sim_{\odot\bar{\mathcal{M}}}} \mid x \in |\bar{\mathcal{M}}[i]|\},$$

$$|\bar{\mathcal{M}}|_{(i<)} \coloneqq (\bigcup_{j=1}^{i} |\bar{\mathcal{M}}|_{(j)}) \setminus |\bar{\mathcal{M}}|_{(i)}, \qquad |\bar{\mathcal{M}}|_{(i>)} \coloneqq (\bigcup_{j=i}^{n} |\bar{\mathcal{M}}|_{(j)}) \setminus |\bar{\mathcal{M}}|_{(i)}.$$

In this section, we introduce the following relation $\longrightarrow_\tau^{\bar{\mathcal{M}},\mathcal{A}}$.

▶ **Definition 12.** *Let $\bar{\mathcal{M}} \in \mathsf{REL}^+$. The relation $p \longrightarrow_\tau^{\bar{\mathcal{M}},\mathcal{A}} q$ (or, $p \longrightarrow_\tau^{\bar{\mathcal{M}}} q$ if $\mathcal{A}$ is clear) is defined as the minimal ternary relation of $p,q \in |\mathcal{A}|$ and an $\odot\bar{\mathcal{M}}$-run τ such that $(1^\tau, 2^\tau) \in \bigcup_{i=1}^n |\bar{\mathcal{M}}|_{(i)}^2$, closed under the following rules:*

$$\frac{p \in |\mathcal{A}| \text{ and } x \in |\bar{\mathcal{M}}[i]|}{p \longrightarrow_{[(i,x)]_\sim}^{\bar{\mathcal{M}}} p}\ R \qquad \frac{(p,q) \in \alpha^{\mathcal{A}} \text{ and } (x,y) \in [\![\alpha]\!]^{\bar{\mathcal{M}}[i]}}{p \longrightarrow_{[(i,x)]_\sim \alpha [(i,y)]_\sim}^{\bar{\mathcal{M}}} q}\ \alpha \text{ for } \alpha \in \Sigma \sqcup \{1\}$$

$$\frac{p \longrightarrow_\tau^{\bar{\mathcal{M}}} q \quad q \longrightarrow_\sigma^{\bar{\mathcal{M}}} r}{p \longrightarrow_{\tau\diamond\sigma}^{\bar{\mathcal{M}}} r}\ T \qquad \frac{(p,q) \in \ell^{\mathcal{A}}_{(p',q')}, p' \longrightarrow_\tau^{\bar{\mathcal{M}}} q', \text{ and } 1^\tau = 2^\tau = [(i,x)]_\sim \in |\odot\bar{\mathcal{M}}|}{p \longrightarrow_{[(i,x)]_\sim \ell_{(p',q')} [(i,x)]_\sim}^{\bar{\mathcal{M}}} q}\ \ell_{(p',q')}$$

The relation $\longrightarrow_\tau^{\bar{\mathcal{M}}}$ is a "decomposed" version of $\longrightarrow_\tau^{\odot\bar{\mathcal{M}}}$ in that runs τ in $\longrightarrow_\tau^{\bar{\mathcal{M}}}$ are constructed by composing runs such that $(1^\tau, 2^\tau) \in \bigcup_{i=1}^n |\bar{\mathcal{M}}|_{(i)}^2$. The following theorem shows that $\longrightarrow_\tau^{\bar{\mathcal{M}}}$ coincides with $\longrightarrow_\tau^{\odot\bar{\mathcal{M}}}$ (under the restriction of runs τ that $(1^\tau, 2^\tau) \in \bigcup_{i=1}^n |\bar{\mathcal{M}}|_{(i)}^2$).

▶ **Theorem 13** (Decomposition theorem). *Let $\mathcal{A}$ be a loop-automaton and let $\bar{\mathcal{M}} \in \mathsf{REL}^+$. For all $p,q \in |\mathcal{A}|$ and runs τ on $\odot\bar{\mathcal{M}}$ such that $(1^\tau, 2^\tau) \in \bigcup_{i=1}^n |\bar{\mathcal{M}}|_{(i)}^2$, we have:*

$$p \longrightarrow_\tau^{\odot\bar{\mathcal{M}}} q \quad \Longleftrightarrow \quad p \longrightarrow_\tau^{\bar{\mathcal{M}}} q.$$

The direction ($\Longleftarrow$) is easy by using the derivation tree of the same form. The converse direction ($\Longrightarrow$) is relatively non-trivial. We first give an example.

▶ **Example 14.** Recall the derivation tree given in Example 9. In the tree, we cannot replace $(\longrightarrow_{\odot\bar{\mathcal{N}}}^{\mathcal{M}})$ with $(\longrightarrow^{\bar{\mathcal{N}}})$ straightforwardly, due to that some runs contain the pair $\tilde{1}$ and $\tilde{3}$ as the source and target (note that there is no i such that $\{\tilde{1},\tilde{3}\} \subseteq |\bar{\mathcal{N}}|_{(i)}$). Nevertheless, we can show $(\tilde{1},\tilde{1}) \in [\![\mathcal{A}]\!]^{\odot\bar{\mathcal{N}}}$ by modifying to the following derivation tree:

$$\dfrac{p_1 \longrightarrow_{\tilde{1}a\tilde{2}}^{\bar{\mathcal{N}}} p_2 \qquad \dfrac{\dfrac{\dfrac{q_1 \longrightarrow_{\tilde{3}b\tilde{2}}^{\bar{\mathcal{N}}} q_2 \quad q_2 \longrightarrow_{\tilde{2}a\tilde{3}}^{\bar{\mathcal{N}}} q_3}{q_1 \longrightarrow_{\tilde{3}b\tilde{2}a\tilde{3}}^{\bar{\mathcal{N}}} q_3}\ \mathrm{T}}{p_3 \longrightarrow_{\tilde{3}\ell_{(q_1,q_3)}\tilde{3}}^{\bar{\mathcal{N}}} p_4}\ \ell_{(q_1,q_3)} \qquad p_4 \longrightarrow_{\tilde{3}b\tilde{2}}^{\bar{\mathcal{N}}} p_5}{p_2 \longrightarrow_{\tilde{2}a\tilde{3}\ell_{(q_1,q_3)}\tilde{3}b\tilde{2}}^{\bar{\mathcal{N}}} p_5}\ \mathrm{T} \qquad p_5 \longrightarrow_{\tilde{2}b\tilde{1}}^{\bar{\mathcal{N}}} p_6}{p_1 \longrightarrow_{\tilde{1}a\tilde{2}a\tilde{3}\ell_{(q_1,q_3)}\tilde{3}b\tilde{2}b\tilde{1}}^{\bar{\mathcal{N}}} p_6}\ \mathrm{T}$$

Proof of Theorem 13

We first observe the following lemma.

▶ **Lemma 15.** *Let $\mathcal{A}$ be a loop-automaton and let $\mathcal{M} \in \mathsf{REL}$. Let $p, q \in |\mathcal{A}|$ and let $\tau = \tilde{x}_0\alpha_1\tilde{x}_1 \dots \tilde{x}_{m-1}\alpha_m\tilde{x}_m$. If $p \longrightarrow_\tau^{\mathcal{M}} q$, then there exist $r_0, \dots, r_m$ such that $r_0 = p$, $r_m = q$, and for each $j \in [m]$, we have $r_{j-1} \longrightarrow^{\mathcal{M}}_{\tilde{x}_{j-1}\alpha_j\tilde{x}_j} r_j$.*

Proof. By easy induction on the derivation tree of $p \longrightarrow_\tau^{\mathcal{M}} q$. ◀

For instance, for $p_1 \longrightarrow^{\odot\bar{\mathcal{N}}}_{\tilde{1}a\tilde{2}a\tilde{3}\ell_{(q_1,q_3)}\tilde{3}b\tilde{2}b\tilde{1}} p_6$ in Example 9, we can take the following sequence: $p_1 \longrightarrow^{\odot\bar{\mathcal{N}}}_{\tilde{1}a\tilde{2}} p_2 \longrightarrow^{\odot\bar{\mathcal{N}}}_{\tilde{2}a\tilde{3}} p_3 \longrightarrow^{\odot\bar{\mathcal{N}}}_{\tilde{3}\ell_{(q_1,q_3)}\tilde{3}} p_4 \longrightarrow^{\odot\bar{\mathcal{N}}}_{\tilde{3}b\tilde{2}} p_5 \longrightarrow^{\odot\bar{\mathcal{N}}}_{\tilde{2}b\tilde{1}} p_6$.

Proof of Theorem 13. ($\Longleftarrow$): By using the derivation tree of the same form. ($\Longrightarrow$): By induction on the length of τ. We distinguish the following cases:

- Case $\tau = \tilde{x}$ and Case $\tau = \tilde{x}\alpha\tilde{y}$ where $\alpha \in L_{\Sigma,|\mathcal{A}|}$: Easy by applying the same rule.
- Case $\tau = \tilde{x}_0\alpha_1\tilde{x}_1 \dots \tilde{x}_{m-1}\alpha_m\tilde{x}_m$ where $m \geq 2$ and $\alpha_i \in L_{\Sigma,|\mathcal{A}|}$ for each $i \in [m]$. For each $j \in [m]$, let $x_j (\in \bar{\mathcal{M}}[i'])$ be such that $\tilde{x}_j = [(i', x_j)]_\sim$ for some i'. Note that $\tilde{x}_0 = [(i, x_0)]_\sim$ and $\tilde{x}_m = [(i, x_m)]_\sim$ holds. By Lemma 15, there exist $r_0, \dots, r_m$ such that $r_0 = p$, $r_m = q$, and $r_{j-1} \longrightarrow^{\odot\bar{\mathcal{M}}}_{\tilde{x}_{j-1}\alpha_j\tilde{x}_j} r_j$ for each j. We distinguish the following cases:
  - Case $\tilde{x}_1 \in |\bar{\mathcal{M}}|_{(i)}$: We then have:

$$\dfrac{\dfrac{}{r_0 \longrightarrow^{\bar{\mathcal{M}}}_{\tilde{x}_0\alpha_1\tilde{x}_1} r_1}\ \alpha_1 \qquad \dfrac{}{r_1 \longrightarrow^{\bar{\mathcal{M}}}_{\tilde{x}_1\alpha_2\dots\alpha_m\tilde{x}_m} r_m}\ \text{IH at } i}{r_0 \longrightarrow^{\bar{\mathcal{M}}}_{\tilde{x}_0\alpha_1\dots\alpha_m\tilde{x}_m} r_m}\ \mathrm{T}$$

  - Case $\tilde{x}_{m-1} \in |\bar{\mathcal{M}}|_{(i)}$: Similar to the case above.
  - Case $(\tilde{x}_1, \tilde{x}_{m-1}) \in |\bar{\mathcal{M}}|^2_{(i<)} \sqcup |\bar{\mathcal{M}}|^2_{(i>)}$: WLOG, we consider when $(\tilde{x}_1, \tilde{x}_{m-1}) \in |\bar{\mathcal{M}}|^2_{(i>)}$. For each $j' \in [m]$, there exists some $i_{j'} > i$ such that $\tilde{x}_{j'-1} = [(i_{j'}, x_{j'-1})]_\sim$, $\tilde{x}_{j'} = [(i_{j'}, x_{j'})]_\sim$, and $(x_{j'-1}, x_{j'}) \in \alpha_{j'}^{\bar{\mathcal{M}}[i_{j'}]}$.
    - Case $i_1 \leq i_{m-1}$: By the definition of $\sim_{\odot\bar{\mathcal{M}}}$ with $\tilde{x}_m = [(i_{m-1}, x_m)]_\sim = [(i, x_m)]_\sim$ and $i \leq i_1 \leq i_{m-1}$, we have $\tilde{x}_m = [(i_1, x_m)]_\sim$. As $\tilde{x}_1, \tilde{x}_m \in |\bar{\mathcal{M}}|_{(i_1)}$, we have:

$$\dfrac{\dfrac{}{r_0 \longrightarrow^{\bar{\mathcal{M}}}_{\tilde{x}_0\alpha_1\tilde{x}_1} r_1}\ \alpha_1 \qquad \dfrac{}{r_1 \longrightarrow^{\bar{\mathcal{M}}}_{\tilde{x}_1\alpha_2\dots\alpha_m\tilde{x}_m} r_m}\ \text{IH at } i_1}{r_0 \longrightarrow^{\bar{\mathcal{M}}}_{\tilde{x}_0\alpha_1\dots\alpha_m\tilde{x}_m} r_m}\ \mathrm{T}$$

    - Case $i_1 \geq i_{m-1}$: Similarly, we have $\tilde{x}_1, \tilde{x}_m \in |\bar{\mathcal{M}}|_{(i_{m-1})}$, and thus we have:

$$\dfrac{\dfrac{}{r_0 \longrightarrow^{\bar{\mathcal{M}}}_{\tilde{x}_0\alpha_1\dots\alpha_{m-1}\tilde{x}_{m-1}} r_{m-1}}\ \text{IH at } i_{m-1} \qquad \dfrac{}{r_{m-1} \longrightarrow^{\bar{\mathcal{M}}}_{\tilde{x}_{m-1}\alpha_m\tilde{x}_m} r_m}\ \alpha_m}{r_0 \longrightarrow^{\bar{\mathcal{M}}}_{\tilde{x}_0\alpha_1\dots\alpha_m\tilde{x}_m} r_m}\ \mathrm{T}$$

  - Otherwise, $(\tilde{x}_1, \tilde{x}_{m-1}) \in (|\bar{\mathcal{M}}|_{(i<)} \times |\bar{\mathcal{M}}|_{(i>)}) \sqcup (|\bar{\mathcal{M}}|_{(i>)} \times |\bar{\mathcal{M}}|_{(i<)})$: We observe that $\{|\bar{\mathcal{M}}|_{(i<)}, |\bar{\mathcal{M}}|_{(i)}, |\bar{\mathcal{M}}|_{(i>)}\}$ is a 3-partition of the set $|\odot\bar{\mathcal{M}}|$ and $(\tilde{x}_{j-1}, \tilde{x}_j) \notin (|\bar{\mathcal{M}}|_{(i<)} \times |\bar{\mathcal{M}}|_{(i>)}) \sqcup (|\bar{\mathcal{M}}|_{(i>)} \times |\bar{\mathcal{M}}|_{(i<)})$ for each j. Then by an analog of the "discrete intermediate value theorem", there exists some $1 < k < m-1$ such that $\tilde{x}_k \in |\bar{\mathcal{M}}|_{(i)}$. We then have:

$$\dfrac{\dfrac{}{r_0 \longrightarrow^{\bar{\mathcal{M}}}_{\tilde{x}_0\alpha_1\dots\alpha_k\tilde{x}_k} r_k}\ \text{IH at } i \qquad \dfrac{}{r_k \longrightarrow^{\bar{\mathcal{M}}}_{\tilde{x}_k\alpha_{k+1}\dots\alpha_m\tilde{x}_m} r_m}\ \text{IH at } i}{r_0 \longrightarrow^{\bar{\mathcal{M}}}_{\tilde{x}_0\alpha_1\tilde{x}_1\dots\tilde{x}_{m-1}\alpha_m\tilde{x}_m} r_m}\ \mathrm{T}$$

Hence, this completes the proof. ◀

6 Automata Construction

In this section, we show that Theorem 13 induces a reduction from the (relational semantics) inclusion of loop-automata to the (string language) inclusion of 2-way alternating finite string automata (2AFAs). Our reduction is based on [27, §5.3], but the main difference is that we apply a binary encoding (§ 6.4) for obtaining the PSPACE upper bound.

6.1 2AFA

We use $\triangleright$ and $\triangleleft$ as the special characters denoting the leftmost and rightmost anchors. A *2AFA* $\mathcal{A}$ over a finite set A is a tuple $\mathcal{A} = (|\mathcal{A}|, \delta^{\mathcal{A}}, 1^{\mathcal{A}})$, where

- $|\mathcal{A}|$ is a finite set of states,
- $\delta^{\mathcal{A}} \colon |\mathcal{A}| \times (A \sqcup \{\triangleright, \triangleleft\}) \to \mathbb{B}_+(|\mathcal{A}| \times \{-1, 0, 1\})$ is a *transition function*, where $\mathbb{B}_+(X)$ denotes the set of *positive boolean formulas* over a set X of *propositional variables* given by

$$\varphi, \psi \in \mathbb{B}_+(X) ::= p \mid \mathsf{false} \mid \mathsf{true} \mid \varphi \vee \psi \mid \varphi \wedge \psi \qquad \text{where } p \in X,$$

- $1^{\mathcal{A}} \in |\mathcal{A}|$ is the initial state.

For a 2AFA $\mathcal{A}$ and a string $w = a_1 \dots a_n$ over $A \sqcup \{\triangleright, \triangleleft\}$, the set $S^{\mathcal{A}}_w \subseteq |\mathcal{A}| \times [n]$ is defined as the smallest set closed under the following rule: For each $(q, i) \in |\mathcal{A}| \times [n]$ and propositional variables $(q_1, i_1), \dots, (q_m, i_m) \in |\mathcal{A}| \times \{-1, 0, 1\}$, when the positive boolean formula $\delta^{\mathcal{A}}(q, a_i)$ is semantically equivalent to true under the assumption that each (q_k, i_k) is true, then

$$\frac{(q_1, i + i_1) \in S^{\mathcal{A}}_w \quad \dots \quad (q_m, i + i_m) \in S^{\mathcal{A}}_w}{(q, i) \in S^{\mathcal{A}}_w}.$$

For a 2AFA $\mathcal{A}$, the language is defined as $[\mathcal{A}] := \{w \in A^* \mid (1^{\mathcal{A}}, 1) \in S^{\mathcal{A}}_{\triangleright w \triangleleft}\}$. We define the *size* $\|\mathcal{A}\|$ as $\sum_{(q,a) \in |\mathcal{A}| \times (A \sqcup \{\triangleright, \triangleleft\})} \|\delta^{\mathcal{A}}(q, a)\|$, where $\|\varphi\|$ denotes the number of symbols occurring in the positive boolean formula φ.

By an on-the-fly exponential-size 1-way NFA construction [12, Lem. 1 and 5] for the language $[\mathcal{A}] \setminus [\mathcal{B}]$ (see also [27, Prop. 5.6]), we have the following proposition:

▶ **Proposition 16.** *The language inclusion problem for 2AFAs – given a finite set A and given two 2AFAs $\mathcal{A}$ and $\mathcal{B}$ over A, does $[\mathcal{A}] \subseteq [\mathcal{B}]$ hold? – is decidable in* PSPACE.

6.2 2AFAs construction

Let $\mathsf{c}_1, \mathsf{c}_2, \dots$ be a set of pairwise distinct elements. We write REL_k for the class of all structures $\mathcal{M}$ such that $|\mathcal{M}| \subseteq \{\mathsf{c}_1, \dots, \mathsf{c}_k\}$. Below, we consider path decompositions of width $k - 1$ as strings over REL_k. Additionally, we only consider *normalized* path decompositions, based on [27, §5.3.3 & 5.3.4]. Let $\mathcal{L}^{\mathrm{Norm}}_k := \mathsf{REL}^+_k \setminus (\mathcal{L}^{\mathrm{Inac}}_k \cup \mathcal{L}^{\mathrm{Incon}}_k)$, where

$$\mathcal{L}^{\mathrm{Inac}}_k := \mathsf{REL}^*_k \,;\, \{\mathcal{M}\mathcal{N} \in \mathsf{REL}^2_k \mid |\mathcal{M}| \cap |\mathcal{N}| = \emptyset\} \,;\, \mathsf{REL}^*_k,$$
$$\mathcal{L}^{\mathrm{Incon}}_k := \mathsf{REL}^*_k \,;\, \{\mathcal{M}\mathcal{N} \in \mathsf{REL}^2_k \mid \exists b \in \Sigma,\ b^{\mathcal{M}} \cap (|\mathcal{M}| \cap |\mathcal{N}|)^2 \neq b^{\mathcal{N}} \cap (|\mathcal{M}| \cap |\mathcal{N}|)^2\} \,;\, \mathsf{REL}^*_k.$$

Here, ; denotes the concatenation of languages. $\mathcal{L}^{\mathrm{Inac}}_k$ denotes the class of path decompositions having an inaccessible pair of vertices under the assumption that every pair of each bag is connected. $\mathcal{L}^{\mathrm{Incon}}_k$ denotes the class of path decompositions having an inconsistent pair of

vertices in that there is an edge between the two vertices in one bag but not in the other bag. $\mathcal{L}_k^{\mathrm{Norm}}$ is sufficient to enumerate all structures of pathwidth at most $k-1$ in that $\mathsf{REL}_{\mathrm{pw}\le k-1}$ coincides with the isomorphism closure of $\{\odot\bar{\mathcal{M}} \mid \bar{\mathcal{M}} \in \mathcal{L}_k^{\mathrm{Norm}}\}$ [27, Proposition 5.11].

Using 2AFAs, we can naturally encode the rules of Definition 12.

▶ **Definition 17.** *For $k \ge 2$ and a loop-automaton $\mathcal{A}$, the 2AFA $\mathcal{B}_k^{\mathcal{A}}$ over REL_k is defined as follows:*

- $|\mathcal{B}_k^{\mathcal{A}}| \coloneqq \{1\} \sqcup (([k] \times |\mathcal{A}|)^2 \times \{?, \checkmark\})$*; we abbreviate* $((x,p),(y,q),m)$ *to* $((x,p),(y,q))_m$*,*
- $\delta^{\mathcal{B}_k^{\mathcal{A}}}(q,a) \coloneqq \bigvee\{\psi \mid (q,a) \rightsquigarrow \psi\}$ *where* $(\rightsquigarrow) \subseteq (|\mathcal{B}_k^{\mathcal{A}}| \times \mathsf{REL}_k) \times \mathbb{B}_+(|\mathcal{B}_k^{\mathcal{A}}| \times \{-1,0,+1\})$ *is the minimal set closed under the following rules:*

$$(1, \triangleright) \rightsquigarrow (((x, 1^{\mathcal{A}}), (y, 2^{\mathcal{A}}))_?, +1) \quad \text{for } x,y \in [k] \tag{I}$$
$$((\tilde{p},\tilde{p})_\checkmark, \mathcal{M}) \rightsquigarrow \mathsf{true} \quad \text{if } \tilde{p} \in |\mathcal{M}| \times |\mathcal{A}| \tag{R}$$
$$((\tilde{p},\tilde{q})_\checkmark, \mathcal{M}) \rightsquigarrow ((\tilde{p},\tilde{r})_\checkmark, 0) \wedge ((\tilde{r},\tilde{q})_\checkmark, 0) \quad \text{if } \tilde{r} \in |\mathcal{M}| \times |\mathcal{A}| \tag{T}$$
$$(((x,p),(y,q))_\checkmark, \mathcal{M}) \rightsquigarrow \mathsf{true} \quad \text{if } (p,q) \in \alpha^{\mathcal{A}} \text{ and } (x,y) \in [\![\alpha]\!]^{\mathcal{M}} \text{ for } \alpha \in \Sigma \sqcup \{1\} \tag{α}$$
$$(((x,p),(x,q))_\checkmark, \mathcal{M}) \rightsquigarrow (((x,p'),(x,q'))_\checkmark, 0) \quad \text{if } (p,q) \in \ell^{\mathcal{A}}_{(p',q')} \tag{$\ell_{(p',q')}$}$$
$$((\tilde{p},\tilde{q})_\checkmark, \mathcal{M}) \rightsquigarrow ((\tilde{p},\tilde{q})_?, +1) \tag{$+1$}$$
$$((\tilde{p},\tilde{q})_\checkmark, \mathcal{M}) \rightsquigarrow ((\tilde{p},\tilde{q})_?, -1) \tag{-1}$$
$$((\tilde{p},\tilde{q})_?, \mathcal{M}) \rightsquigarrow ((\tilde{p},\tilde{q})_\checkmark, 0) \quad \text{if } \tilde{p},\tilde{q} \in |\mathcal{M}| \times |\mathcal{A}| \tag{$\checkmark$}$$

- $1^{\mathcal{B}_k^{\mathcal{A}}} \coloneqq 1$.

The rules are defined based on those in Definition 12, where

- the rules $(+1)$ and (-1) are used for moving vertices in the same quotient class;
- the check mark "$\checkmark$" in $((x,p),(y,q))_\checkmark$ expresses that $x,y \in |\mathcal{M}|$ holds (where $\mathcal{M}$ is the structure in the current position); the question mark "?" in $((x,p),(y,q))_?$ expresses that it has not yet been checked. They are introduced to check it after the rule $(+1)$ or (-1).

By construction of $\mathcal{B}_k^{\mathcal{A}}$, we can show the following. Both directions are shown by easy induction on the derivative trees (cf., [27, Proposition 5.9]).

▶ **Proposition 18.** *Let $k \ge 2$ and let $\mathcal{A}$ be a loop-automaton. Let $\bar{\mathcal{M}} \in \mathsf{REL}_k^+$. For all $j \in [n]$, $x,y \in |\bar{\mathcal{M}}[j]|$, and $p,q \in |\mathcal{A}|$, the following are equivalent:*

- $p \longrightarrow_\tau^{\bar{\mathcal{M}}} q$ *for some run τ of $\mathcal{A}$ s.t. $1^\tau = [(j,x)]_\sim$ and $2^\tau = [(j,y)]_\sim$;*
- $(((x,p),(y,q))_\checkmark, j) \in S^{\mathcal{B}_k^{\mathcal{A}}}_{\triangleright\bar{\mathcal{M}}\triangleleft}$.

For a set $\mathcal{L} \subseteq \mathsf{REL}_k^+$, we say that a loop-automaton $\mathcal{A}$ is *closed* on $\mathcal{L}$ if $[\![\mathcal{A}]\!]^{\odot\bar{\mathcal{M}}} = \emptyset$ or $[\![\mathcal{A}]\!]^{\odot\bar{\mathcal{M}}} = |\odot\bar{\mathcal{M}}|^2$ holds for every $\bar{\mathcal{M}} \in \mathcal{L}$. By Proposition 18, we have the following.

▶ **Lemma 19.** *For $k \ge 2$ and a loop-automaton $\mathcal{A}$, we have:*

$$[\mathcal{B}_k^{\mathcal{A}}] = \{\bar{\mathcal{M}} \in \mathsf{REL}_k^+ \mid ([(1,x)]_\sim, [(1,y)]_\sim) \in [\![\mathcal{A}]\!]^{\odot\bar{\mathcal{M}}} \text{ for some } x,y \in |\bar{\mathcal{M}}[1]|\}.$$

In particular, for a set $\mathcal{L} \subseteq \mathsf{REL}_k^+$, if $\mathcal{A}$ is closed on $\mathcal{L}$, we have:

$$[\mathcal{B}_k^{\mathcal{A}}] \cap \mathcal{L} = \{\bar{\mathcal{M}} \in \mathcal{L} \mid [\![\mathcal{A}]\!]^{\odot\bar{\mathcal{M}}} \neq \emptyset\}.$$

Proof. We have:

$$
\begin{aligned}
&\bar{\mathcal{M}} \in [\mathcal{B}_k^{\mathcal{A}}] \\
&\Leftrightarrow \exists x, y \in |\bar{\mathcal{M}}[1]|,\ (((x, 1^{\mathcal{A}}), (y, 2^{\mathcal{A}}))_{\checkmark}, 1) \in S^{\mathcal{B}_k^{\mathcal{A}}}_{\triangleright \bar{\mathcal{M}} \triangleleft} && \text{(By the form of } \mathcal{B}_k^{\mathcal{A}}) \\
&\Leftrightarrow \exists x, y \in |\bar{\mathcal{M}}[1]|, \exists \tau \text{ s.t. } 1^{\tau} = [(1, x)]_{\sim} \text{ and } 2^{\tau} = [(1, y)]_{\sim},\ 1^{\mathcal{A}} \longrightarrow_{\tau}^{\bar{\mathcal{M}}} 2^{\mathcal{A}} \\
& && \text{(Proposition 18)} \\
&\Leftrightarrow \exists x, y \in |\bar{\mathcal{M}}[1]|, \exists \tau \text{ s.t. } 1^{\tau} = [(1, x)]_{\sim} \text{ and } 2^{\tau} = [(1, y)]_{\sim},\ 1^{\mathcal{A}} \longrightarrow_{\tau}^{\odot\bar{\mathcal{M}}} 2^{\mathcal{A}} \\
& && \text{(Theorem 13)} \\
&\Leftrightarrow \exists x, y \in |\bar{\mathcal{M}}[1]|,\ ([(1, x)]_{\sim}, [(1, y)]_{\sim}) \in [\![\mathcal{A}]\!]^{\odot\bar{\mathcal{M}}}. && \text{(By definition)}
\end{aligned}
$$

Hence, this completes the proof. (The second claim is immediate from the first one.) ◀

6.3 Encoding the equational theory

Let $c_{\top}$ be a variable for encoding the top element $\top$ and let l and r be variables for indicating the left and right vertex, respectively. We define

$$\mathcal{L}_k^{\mathrm{Top}} \coloneqq \mathsf{REL}_k^* \,;\, \{\mathcal{M} \in \mathsf{REL}_k \mid c_{\top}^{\mathcal{M}} \neq |\mathcal{M}|^2\} \,;\, \mathsf{REL}_k^*.$$

For every $\bar{\mathcal{M}} \in \mathcal{L}_k^{\mathrm{Norm}} \setminus \mathcal{L}_k^{\mathrm{Top}}$, we can encode the top element $\top$ by the term $c_{\top}^*$: *i.e.*, $[\![c_{\top}^*]\!]^{\odot\bar{\mathcal{M}}} = |\odot\bar{\mathcal{M}}|^2$ holds. Using them, we can encode the equational theory as follows. (Below, we recall $\mathcal{A}_t$ in Definition 10.)

▶ **Lemma 20.** *Let t and s be terms such that $\mathsf{c}_{\top}, l, r$ do not occur. Then,*

$$\mathsf{REL}_{\mathrm{pw}\leq k-1} \models t \leq s \quad \Longleftrightarrow \quad [\mathcal{B}_k^{\mathcal{A}_{c_{\top}^* ltrc_{\top}^*}}] \subseteq [\mathcal{B}_k^{\mathcal{A}_{c_{\top}^* lsrc_{\top}^*}}] \cup \mathcal{L}_k^{\mathrm{Inac}} \cup \mathcal{L}_k^{\mathrm{Incon}} \cup \mathcal{L}_k^{\mathrm{Top}}.$$

Proof. We have:

$$
\begin{aligned}
&\mathsf{REL}_{\mathrm{pw}\leq k-1} \models t \leq s \ \Leftrightarrow\ \{\odot\bar{\mathcal{M}} \mid \bar{\mathcal{M}} \in \mathcal{L}_k^{\mathrm{Norm}}\} \models t \leq s \\
&\quad\Leftrightarrow\ \{\odot\bar{\mathcal{M}} \mid \bar{\mathcal{M}} \in \mathcal{L}_k^{\mathrm{Norm}} \setminus \mathcal{L}_k^{\mathrm{Top}}\} \models t \leq s && (c_{\top} \text{ is fresh}) \\
&\quad\Leftrightarrow\ \{\odot\bar{\mathcal{M}} \mid \bar{\mathcal{M}} \in \mathcal{L}_k^{\mathrm{Norm}} \setminus \mathcal{L}_k^{\mathrm{Top}}\} \models c_{\top}^* l t r c_{\top}^* \leq c_{\top}^* l s r c_{\top}^* && (c_{\top}^* \text{ expresses } \top;\ l, r \text{ are fresh}) \\
&\quad\Leftrightarrow\ \forall \bar{\mathcal{M}} \in \mathcal{L}_k^{\mathrm{Norm}} \setminus \mathcal{L}_k^{\mathrm{Top}},\ [\![\mathcal{A}_{c_{\top}^* ltrc_{\top}^*}]\!]^{\odot\bar{\mathcal{M}}} \subseteq [\![\mathcal{A}_{c_{\top}^* lsrc_{\top}^*}]\!]^{\odot\bar{\mathcal{M}}} && \text{(Proposition 11)} \\
&\quad\Leftrightarrow\ \{\bar{\mathcal{M}} \mid [\![\mathcal{A}_{c_{\top}^* ltrc_{\top}^*}]\!]^{\odot\bar{\mathcal{M}}} \neq \emptyset\} \cap (\mathcal{L}_k^{\mathrm{Norm}} \setminus \mathcal{L}_k^{\mathrm{Top}}) \subseteq \{\bar{\mathcal{M}} \mid [\![\mathcal{A}_{c_{\top}^* lsrc_{\top}^*}]\!]^{\odot\bar{\mathcal{M}}} \neq \emptyset\} \\
&\qquad\qquad (\mathcal{A}_{c_{\top}^* ltrc_{\top}^*} \text{ and } \mathcal{A}_{c_{\top}^* lsrc_{\top}^*} \text{ are closed on } \mathcal{L}_k^{\mathrm{Norm}} \setminus \mathcal{L}_k^{\mathrm{Top}}) \\
&\quad\Leftrightarrow\ [\mathcal{B}_k^{\mathcal{A}_{c_{\top}^* ltrc_{\top}^*}}] \cap (\mathcal{L}_k^{\mathrm{Norm}} \setminus \mathcal{L}_k^{\mathrm{Top}}) \subseteq [\mathcal{B}_k^{\mathcal{A}_{c_{\top}^* lsrc_{\top}^*}}] && \text{(Lemma 19)} \\
&\quad\Leftrightarrow\ [\mathcal{B}_k^{\mathcal{A}_{c_{\top}^* ltrc_{\top}^*}}] \subseteq [\mathcal{B}_k^{\mathcal{A}_{c_{\top}^* lsrc_{\top}^*}}] \cup \mathcal{L}_k^{\mathrm{Inac}} \cup \mathcal{L}_k^{\mathrm{Incon}} \cup \mathcal{L}_k^{\mathrm{Top}}.
\end{aligned}
$$

Hence, this completes the proof. ◀

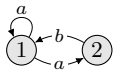
6.4 Binary encoding of structures

The size of $\#\mathsf{REL}_k$ is approximately $2^{\#\Sigma \cdot k^2}$. Due to this, the size $\|\mathcal{B}_k^{\mathcal{A}}\|$ is exponential in k. Nevertheless, we can avoid this exponential blowup by a standard binary encoding of structures (cf., e.g., [16, Definition 2.1]).

Let $\Sigma = \{a_1, \ldots, a_m\}$. Given $k \geq 2$ and a structure $\mathcal{M} \in \mathsf{REL}_k$, we define the following $(k + mk^2)$-bit string $\lfloor \mathcal{M} \rfloor_k$:

- The first k bits encode the universe $\lfloor\mathcal{M}\rfloor \subseteq [k]$: the i-th bit is 1 if $i \in \lfloor\mathcal{M}\rfloor$, and 0 otherwise.
- For each ℓ from 1 to m, the next k^2 bits encode the relation $a_\ell^{\mathcal{M}} \subseteq \lfloor\mathcal{M}\rfloor^2$: the $((p-1)k+q)$-th bit is 1 if $(p,q) \in a_\ell^{\mathcal{M}}$, and 0 otherwise.

For instance, if $\mathcal{M}$ is given as follows, $k=3$, and $\Sigma=\{a,b\}$, then $\lfloor\mathcal{M}\rfloor_k$ is defined as follows:

$\mathcal{M}$ =  $\rightsquigarrow$

$\lfloor\mathcal{M}\rfloor_k$ =	$\lvert\mathcal{M}\rvert$			$a^{\mathcal{M}}$								$b^{\mathcal{M}}$								
	1	1	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

For a sequence $\bar{\mathcal{M}} = \mathcal{M}_1 \dots \mathcal{M}_n \in \mathsf{REL}_k^+$, we also define the string $\lfloor\bar{\mathcal{M}}\rfloor_k \coloneqq \lfloor\mathcal{M}_1\rfloor_k \dots \lfloor\mathcal{M}_n\rfloor_k$. For a set $\mathcal{L} \subseteq \mathsf{REL}_k^+$, we also define the set $\lfloor\mathcal{L}\rfloor_k \coloneqq \{\lfloor\bar{\mathcal{M}}\rfloor_k \mid \bar{\mathcal{M}} \in \mathcal{L}\}$. We can redefine the construction of Definition 17 via this binary encoding.

▶ **Lemma 21.** *Given $k \geq 2$ and a loop-automaton $\mathcal{A}$, we can construct a 2AFA $\tilde{\mathcal{B}}_k^{\mathcal{A}}$ over the alphabet $\{0,1\}$ such that $[\tilde{\mathcal{B}}_k^{\mathcal{A}}] = \lfloor[\mathcal{B}_k^{\mathcal{A}}]\rfloor_k$, in polynomial time in k and $\|\mathcal{A}\|$.*

Proof Sketch. From the construction of Definition 17, by a straightforward modification based on the binary encoding above. For instance, for the rule (a_i) (where $a_i \neq \mathsf{I}$), when $(p,q) \in a_i^{\mathcal{A}}$, we replace the rule

$$(((x,p),(y,q))_\checkmark, \mathcal{M}) \rightsquigarrow \mathsf{true} \quad \text{if } (x,y) \in a_i^{\mathcal{M}}$$

with the rules so that we move the head to the position of the bit corresponding to $(x,y) \in a_i^{\mathcal{M}}$ and check whether it is 1; more precisely, as follows:

$$\begin{aligned}
(((x,p),(y,q))_\checkmark, b) &\rightsquigarrow (((x,p),(y,q))_\checkmark^{(a_i),0}, 0), \\
(((x,p),(y,q))_\checkmark^{(a_i),\ell-1}, b) &\rightsquigarrow (((x,p),(y,q))_\checkmark^{(a_i),\ell}, +1) \text{ for } \ell \in [k+mk^2], \\
(((x,p),(y,q))_\checkmark^{(a_i),\ell}, 1) &\rightsquigarrow \mathsf{true} \qquad \text{if } \ell = k + (i-1)k^2 + (x-1)k + y,
\end{aligned}$$

where $b \in \{0,1\}$. Such a construction is done in polynomial time. Additionally, we can also construct a 2AFA recognizing the language

$$\lfloor\mathsf{REL}_k^+\rfloor_k = (\{0,1\}^{k+mk^2} \setminus \bigcup_{\substack{i,i' \in [k] \\ j \in [m]}} \underbrace{\{0,1\}^{i-1}0\{0,1\}^{(k-i)}}_{\text{“}\mathsf{c}_i \notin \mathcal{M}\text{”}} \underbrace{\{0,1\}^{(j-1)k^2}(\{0,1\}^{(i-1)k+i'-1} \cup \{0,1\}^{(i'-1)k+i-1})1}_{\text{“}(\mathsf{c}_i,\mathsf{c}_{i'}) \in a_j^{\mathcal{M}} \text{ or } (\mathsf{c}_{i'},\mathsf{c}_i) \in a_j^{\mathcal{M}}\text{”}} \{0,1\}^*)^+$$

in polynomial time. By taking the intersection of them, we obtain the desired 2AFA. ◀

▶ **Theorem 22.** *The (in)equational theory of loop-RKA – given a finite set Σ and two loop-RKA terms t and s over Σ, does $\mathsf{REL} \models t \leq s$ hold? – is* PSPACE*-complete.*

Proof.

Lower bound. Because the equational theory of RKA, which is equivalent to the language equivalence problem of regular expressions (Footnote 1), is already PSPACE-hard [23].

Upper bound. By letting $k \coloneqq \mathrm{iw}(t)+1$, we have:

$$\begin{aligned}
\mathsf{REL} \models t \leq s &\Leftrightarrow \mathsf{REL}_{\mathrm{pw} \leq k-1} \models t \leq s && \text{(Proposition 7)} \\
&\Leftrightarrow [\mathcal{B}_k^{\mathcal{A}_{c_\top^* ltrc_\top^*}}] \subseteq [\mathcal{B}_k^{\mathcal{A}_{c_\top^* lsrc_\top^*}}] \cup \mathcal{L}_k^{\mathrm{Inac}} \cup \mathcal{L}_k^{\mathrm{Incon}} \cup \mathcal{L}_k^{\mathrm{Top}} && \text{(Lemma 20)} \\
&\Leftrightarrow [\tilde{\mathcal{B}}_k^{\mathcal{A}_{c_\top^* ltrc_\top^*}}] \subseteq [\tilde{\mathcal{B}}_k^{\mathcal{A}_{c_\top^* lsrc_\top^*}}] \cup \lfloor\mathcal{L}_k^{\mathrm{Inac}} \cup \mathcal{L}_k^{\mathrm{Incon}} \cup \mathcal{L}_k^{\mathrm{Top}}\rfloor_k. && \text{(Lemma 21)}
\end{aligned}$$

Here, we can construct 2AFAs for $\lfloor\mathcal{L}_k^{\mathrm{Inac}}\rfloor_k$, $\lfloor\mathcal{L}_k^{\mathrm{Incon}}\rfloor_k$, and $\lfloor\mathcal{L}_k^{\mathrm{Top}}\rfloor_k$ in polynomial time. Thus we can construct a 2AFA for the right-hand side language by taking the union of them. Hence, by Proposition 16, this completes the proof. ◀

The upper bound of Theorem 1 immediately follows from Theorem 22. The lower bound of Theorem 1 also follows from the same proof as in Theorem 22.

7 Loop-RKA with Additional Operators

In this section, we briefly give some encodings of additional operators in our automata construction, based on [27].

For top

Similar to § 6.3, to encode top, it suffices to force the class of structures $\mathcal{M}$ so that

$$[\![c_\top^*]\!]^{\mathcal{M}} = |\mathcal{M}|^2, \tag{top}$$

where $c_\top$ is a fresh variable. For a class $\mathcal{C} \subseteq \mathsf{REL}$, let $\mathcal{C}^{\mathrm{Top}} \coloneqq \{\mathcal{M} \in \mathcal{C} \mid \mathcal{M} \text{ satisfies (top)}\}$. In this class, we can define the top element $\top$ (such that $[\![\top]\!]^{\mathcal{M}} = |\mathcal{M}|^2$ for all $\mathcal{M} \in \mathcal{C}^{\mathrm{Top}}$) by letting $\top$ be $c_\top^*$. To encode the condition (top), we recall the language $\mathcal{L}_k^{\mathrm{Top}}$ (§ 6.3). Then, $\mathsf{REL}^{\mathrm{Top}}_{\mathrm{pw}\le k-1} = \{\odot\bar{\mathcal{M}} \mid \bar{\mathcal{M}} \in \mathsf{REL}_k^+ \setminus (\mathcal{L}_k^{\mathrm{Top}} \cup \mathcal{L}_k^{\mathrm{Incon}} \cup \mathcal{L}_k^{\mathrm{Inac}})\}$. For the language $\lfloor \mathcal{L}_k^{\mathrm{Top}} \rfloor_k$, we can construct a 1NFA of the language such that the number of states is in polynomial time in k and $\#\Sigma$. Hence, we can encode top.

For tests

Let $B, \bar{B} \subseteq \Sigma$ be two disjoint sets of variables with a bijection $\bar{\ } : B \to \bar{B}$. To encode *tests* as in KAT [19], it suffices to force the class of structures $\mathcal{M}$ so that

$$\bar{b}^{\mathcal{M}} = \triangle_{|\mathcal{M}|} \setminus b^{\mathcal{M}} \quad \text{for every } b \in B. \tag{test}$$

For a class $\mathcal{C} \subseteq \mathsf{REL}$, let $\mathcal{C}^{\mathrm{Test}} \coloneqq \{\mathcal{M} \in \mathcal{C} \mid \mathcal{M} \text{ satisfies (test)}\}$. In this class, for $\{\mathsf{I}, \mathsf{0}, ;, +\}$-terms p, we define the complemented term p^- (w.r.t. the identity relation) as follows:

$$b^- \coloneqq \bar{b}, \quad \bar{b}^- \coloneqq b, \quad \mathsf{I}^- \coloneqq \mathsf{0}, \quad \mathsf{0}^- \coloneqq \mathsf{I}, \quad (p\,;q)^- \coloneqq p^- + q^-, \quad (p+q)^- \coloneqq p^-\,;q^-.$$

Then $[\![p^-]\!]^{\mathcal{M}} = \triangle_{|\mathcal{M}|} \setminus [\![p]\!]^{\mathcal{M}}$ holds for all $\mathcal{M} \in \mathsf{REL}^{\mathrm{Test}}$. We thus can encode tests (propositional formulas), by expressing true, false, conjunction, disjunction, as I, $\mathsf{0}$, ;, and $+$, respectively, and expressing complement as above. Using tests, for instance, we can encode propositional while programs [11, 18]:

$$\textbf{while } p \textbf{ do } t \coloneqq (pt)^* p^-, \qquad \textbf{if } p \textbf{ then } t \textbf{ else } s \coloneqq (pt) + (p^- s).$$

To encode the condition (test), we define the following language:

$$\mathcal{L}_k^{\mathrm{Test}} \coloneqq \mathsf{REL}_k^*\,;\{\mathcal{M} \in \mathsf{REL}_k \mid \exists b \in B,\ b^{\mathcal{M}} \sqcup \bar{b}^{\mathcal{M}} \neq \triangle_{|\mathcal{M}|}\}\,;\mathsf{REL}_k^*.$$

Then, $\mathsf{REL}^{\mathrm{Test}}_{\mathrm{pw}\le k-1} = \{\odot\bar{\mathcal{M}} \mid \bar{\mathcal{M}} \in \mathsf{REL}_k^+ \setminus (\mathcal{L}_k^{\mathrm{Test}} \cup \mathcal{L}_k^{\mathrm{Incon}})\}$. For the language $\lfloor \mathcal{L}_k^{\mathrm{Test}} \rfloor_k$, we can construct a 1NFA of the language such that the number of states is in polynomial time in k and $\#\Sigma$. Hence, we can encode tests.

For converse

Let $C, \breve{C} \subseteq \Sigma$ be two disjoint sets of variables with a bijection $\breve{\ } : C \to \breve{C}$. To encode *converse*, it suffices to force the class of structures $\mathcal{M}$ so that

$$\breve{c}^{\mathcal{M}} = (c^{\mathcal{M}})^\smile \quad \text{for every } c \in C. \tag{converse}$$

For a class $\mathcal{C} \subseteq \mathsf{REL}$, let $\mathcal{C}^{\mathrm{Conv}} := \{\mathcal{M} \in \mathcal{C} \mid \mathcal{M} \text{ satisfies (converse)}\}$. In this class, for each term t, we can define a term $t^\smile$ that $[\![t^\smile]\!]^{\mathcal{M}} = ([\![t]\!]^{\mathcal{M}})^\smile$ for all $\mathcal{M} \in \mathsf{REL}^{\mathrm{Conv}}$, as follows:

$$c^\smile := \breve{c}, \quad \breve{c}^\smile := c, \quad \mathsf{I}^\smile := \mathsf{I}, \quad \mathsf{0}^\smile := \mathsf{0}, \quad (t\,;s)^\smile := s^\smile\,;t^\smile,$$
$$(t+s)^\smile := t^\smile + s^\smile, \quad (t^*)^\smile := (t^\smile)^*, \quad (t^{\circlearrowleft})^\smile := (t^\smile)^{\circlearrowleft}.$$

To encode the condition (converse), we define the following language:

$$\mathcal{L}_k^{\mathrm{Conv}} := \mathsf{REL}_k^*\,;\{\mathcal{M} \in \mathsf{REL}_k \mid \exists c \in C,\ (c^{\mathcal{M}})^\smile \neq \breve{c}^{\mathcal{M}}\}\,;\mathsf{REL}_k^*.$$

Then, $\mathsf{REL}_{\mathrm{pw}\leq k-1}^{\mathrm{Conv}} = \{\odot\bar{\mathcal{M}} \mid \bar{\mathcal{M}} \in \mathsf{REL}_k^+ \setminus \mathcal{L}_k^{\mathrm{Conv}}\}$. For the language $\lfloor \mathcal{L}_k^{\mathrm{Conv}} \rfloor_k$, we can construct a 1NFA of the language such that the number of states is in polynomial time in k and $\#\Sigma$. Hence, we can encode converse.

For nominals

Let $L \subseteq \Sigma$ be a set of variables. To encode *nominals* (from hybrid modal logic) [2], it suffices to force the class of structures $\mathcal{M}$ so that

$$\#l^{\mathcal{M}} = 1 \quad \text{for every } l \in L. \tag{nominal}$$

For a class $\mathcal{C} \subseteq \mathsf{REL}$, let $\mathcal{C}^{\mathrm{Nom}} := \{\mathcal{M} \in \mathcal{C} \mid \mathcal{M} \text{ satisfies (nominal)}\}$. To encode the condition (nominal), we define the following language:

$$\mathcal{L}_k^{\mathrm{Nom}} := \bigcup_{l \in L} \left(\begin{array}{l} \{\mathcal{M} \in \mathsf{REL}_k \mid l^{\mathcal{M}} = \emptyset\}^+ \cup (\mathsf{REL}_k^*\,;\{\mathcal{M} \in \mathsf{REL}_k \mid l^{\mathcal{M}} \not\subseteq \triangle_{|\mathcal{M}|}\}\,;\mathsf{REL}_k^*) \cup \\ \mathsf{REL}_k^*\,;\left\{ \begin{array}{l} \mathcal{M}_1 \ldots \mathcal{M}_n \in \mathsf{REL}_k^+ \mid \exists x, \exists y, (x,x) \in l^{\mathcal{M}_1} \\ \wedge\, (y,y) \in l^{\mathcal{M}_n} \wedge (x \neq y \vee \exists j \in [n], x \notin |\mathcal{M}_j|) \end{array} \right\}\,;\mathsf{REL}_k^* \end{array} \right).$$

Then, $\mathsf{REL}_{\mathrm{pw}\leq k-1}^{\mathrm{Nom}} = \{\odot\bar{\mathcal{M}} \mid \bar{\mathcal{M}} \in \mathsf{REL}_k^+ \setminus \mathcal{L}_k^{\mathrm{Nom}}\}$ holds. (By the first line, $l^{\odot\bar{\mathcal{M}}} \neq \emptyset$ and $l^{\odot\bar{\mathcal{M}}} \subseteq \triangle_{|\odot\bar{\mathcal{M}}|}$. By the second line, $\#\{x \mid (x,x) \in l^{\odot\bar{\mathcal{M}}}\} \leq 1$.) For the language $\lfloor \mathcal{L}_k^{\mathrm{Nom}} \rfloor_k$, we can construct a 1NFA of the language such that the number of states is in polynomial time in k and $\#\Sigma$. Hence, we can encode nominals.

Putting it all together

We finally consider loop-RKA with tests, converse, and nominals; precisely, we consider loop-RKA over the class $\mathsf{REL}^{\mathrm{Top,Conv,Test,Nom}}$ via the above encodings for these extensions, where $B, \bar{B}, C, \breve{C}, L$ are pairwise disjoint subsets of Σ. We recall the fact that loop-RKA with tests, converse, and nominals still has the linearly bounded pathwidth model property:

▶ **Proposition 23** ([27, Proposition 6.5]). *For all loop-RKA terms t and s, we have:*

$$\mathsf{REL}^{\mathrm{Top,Conv,Test,Nom}} \models t \leq s \quad \Leftrightarrow \quad \mathsf{REL}_{\mathrm{pw}\leq \mathrm{iw}(t)+\#L}^{\mathrm{Top,Conv,Test,Nom}} \models t \leq s.$$

▶ **Theorem** (Restatement of Theorem 2). *The equational theory for loop-RKA with top, tests, converse, and nominals is* PSPACE*-complete.*

Proof.

Lower bound. Immediate from the lower bound of Theorem 1.

Upper bound. Given two loop-RKA terms with top, tests, converse, and nominals, t and s, let t' and s' be the loop-RKA terms obtained from t and s by replacing top, tests, and converse with their encodings as above, respectively. By letting $k \coloneqq \mathrm{iw}(t) + \#L + 1$, we have:

$$\begin{aligned}
&\mathsf{REL}^{\mathrm{Top,Conv,Test,Nom}} \models t \leq s \;\Leftrightarrow\; \mathsf{REL}^{\mathrm{Top,Conv,Test,Nom}} \models t' \leq s' \\
&\Leftrightarrow\; \mathsf{REL}^{\mathrm{Top,Conv,Test,Nom}}_{\mathrm{pw}\leq k-1} \models t' \leq s' && \text{(Proposition 23)} \\
&\Leftrightarrow\; [\mathcal{B}_k^{\mathcal{A}_{c^*_\top lt' rc^*_\top}}] \subseteq [\mathcal{B}_k^{\mathcal{A}_{c^*_\top ls' rc^*_\top}}] \cup \mathcal{L}_k^{\mathrm{Inac}} \cup \mathcal{L}_k^{\mathrm{Incon}} \cup \mathcal{L}_k^{\mathrm{Top}} \cup \mathcal{L}_k^{\mathrm{Conv}} \cup \mathcal{L}_k^{\mathrm{Test}} \cup \mathcal{L}_k^{\mathrm{Nom}} \\
& && \text{(Lemma 20 with the encodings above)} \\
&\Leftrightarrow\; [\tilde{\mathcal{B}}_k^{\mathcal{A}_{c^*_\top lt' rc^*_\top}}] \subseteq [\tilde{\mathcal{B}}_k^{\mathcal{A}_{c^*_\top ls' rc^*_\top}}] \cup \lfloor \mathcal{L}_k^{\mathrm{Inac}} \cup \mathcal{L}_k^{\mathrm{Incon}} \cup \mathcal{L}_k^{\mathrm{Top}} \cup \mathcal{L}_k^{\mathrm{Conv}} \cup \mathcal{L}_k^{\mathrm{Test}} \cup \mathcal{L}_k^{\mathrm{Nom}} \rfloor_k. \\
& && \text{(Lemma 21)}
\end{aligned}$$

For the right-hand side language, since we can construct the 2AFA for each language in polynomial time, we can construct its 2AFA by taking the union. Hence, by Proposition 16, this completes the proof. ◀

Corollary 3 immediately follows from Theorem 2 via the encoding of domain (d/r-by-$\circlearrowleft$,$\top$).

References

1. 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.
2. Carlos Areces and Balder ten Cate. Hybrid logics. In *Handbook of Modal Logic*, volume 3 of *Studies in Logic and Practical Reasoning*, pages 821–868. Elsevier, 2007. doi:10.1016/S1570-2464(07)80017-6.
3. Paul Brunet and Damien Pous. Kleene algebra with converse. In *International Conference on Relational and Algebraic Methods in Computer Science (RAMiCS)*, volume 8428 of *LNCS*, pages 101–118. Springer, 2014. doi:10.1007/978-3-319-06251-8_7.
4. Paul Brunet and Damien Pous. Petri automata. *Logical Methods in Computer Science*, 13(3), 2017. doi:10.23638/LMCS-13(3:33)2017.
5. Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi. Rewriting of regular expressions and regular path queries. In *ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (PODS)*, pages 194–204. ACM, 1999. doi:10.1145/303976.303996.
6. Bruno Courcelle and Joost Engelfriet. *Graph Structure and Monadic Second-Order Logic*. Number 138 in Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2012. doi:10.1017/CBO9780511977619.
7. Ryszard Danecki. Propositional dynamic logic with strong loop predicate. In *International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 176 of *LNCS*, pages 573–581. Springer, 1984. doi:10.1007/BFb0030342.
8. Jules Desharnais, Bernhard Möller, and Georg Struth. Kleene algebra with domain. *ACM Transactions on Computational Logic*, 7(4):798–833, 2006. doi:10.1145/1183278.1183285.
9. Jules Desharnais and Georg Struth. Modal semirings revisited. In *Mathematics of Program Construction (MPC)*, volume 5133 of *LNTCS*, pages 360–387. Springer, 2008. doi:10.1007/978-3-540-70594-9_19.
10. Jules Desharnais and Georg Struth. Internal axioms for domain semirings. *Science of Computer Programming*, 76(3):181–203, 2011. doi:10.1016/j.scico.2010.05.007.
11. Michael J. Fischer and Richard E. Ladner. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, 18(2):194–211, 1979. doi:10.1016/0022-0000(79)90046-1.

12 Viliam Geffert and Alexander Okhotin. Transforming two-way alternating finite automata to one-way nondeterministic automata. In *International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 8634 of *LNTCS*, pages 291–302. Springer, 2014. doi:10.1007/978-3-662-44522-8_25.

13 Stefan Göller, Markus Lohrey, and Carsten Lutz. PDL with intersection and converse: satisfiability and infinite-state model checking. *The Journal of Symbolic Logic*, 74(1):279–314, 2009. doi:10.2178/jsl/1231082313.

14 Chris Hardin and Dexter Kozen. On the complexity of the Horn theory of REL. Technical report, Cornell University, 2003. URL: https://hdl.handle.net/1813/5612.

15 Robin Hirsch, Marcel Jackson, and Szabolcs Mikulás. The algebra of functions with antidomain and range. *Journal of Pure and Applied Algebra*, 220(6):2214–2239, 2016. doi:10.1016/j.jpaa.2015.11.003.

16 Neil Immerman. *Descriptive Complexity*. Texts in Computer Science. Springer, 1 edition, 1999. doi:10.1007/978-1-4612-0539-5.

17 Dexter Kozen. A completeness theorem for Kleene algebras and the algebra of regular events. *Information and Computation*, 110(2):366–390, 1994. doi:10.1006/inco.1994.1037.

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

19 Dexter Kozen and Frederick Smith. Kleene algebra with tests: Completeness and decidability. In *EACSL Annual Conference on Computer Science Logic (CSL)*, volume 1258 of *LNCS*, pages 244–259. Springer, 1996. doi:10.1007/3-540-63172-0_43.

20 Carsten Lutz and Dirk Walther. PDL with negation of atomic programs. *Journal of Applied Non-Classical Logics*, 15(2):189–213, 2005. doi:10.3166/jancl.15.189-213.

21 Brett McLean. Free Kleene algebras with domain. *Journal of Logical and Algebraic Methods in Programming*, 117:100606, 2020. doi:10.1016/j.jlamp.2020.100606.

22 Robert McNaughton and Hisao Yamada. Regular expressions and state graphs for automata. *IRE Transactions on Electronic Computers*, EC-9(1):39–47, 1960. doi:10.1109/TEC.1960.5221603.

23 Albert R. Meyer and Larry J. Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential space. In *Annual Symposium on Switching and Automata Theory (SWAT)*, pages 125–129. IEEE, 1972. doi:10.1109/SWAT.1972.29.

24 Yoshiki Nakamura. Partial derivatives on graphs for Kleene allegories. In *Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–12. IEEE, 2017. doi:10.1109/LICS.2017.8005132.

25 Yoshiki Nakamura. Existential calculi of relations with transitive closure: Complexity and edge saturations. In *Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–13. IEEE, 2023. doi:10.1109/LICS56636.2023.10175811.

26 Yoshiki Nakamura. Undecidability of the positive calculus of relations with transitive closure and difference: Hypothesis elimination using graph loops. In *International Conference on Relational and Algebraic Methods in Computer Science (RAMiCS)*, volume 14787 of *LNCS*, pages 207–224. Springer, 2024. doi:10.1007/978-3-031-68279-7_13.

27 Yoshiki Nakamura. Derivatives on graphs for the positive calculus of relations with transitive closure. *Logical Methods in Computer Science*, Volume 21, Issue 4, 2025. doi:10.46298/lmcs-21(4:27)2025.

28 Yoshiki Nakamura. Undecidability of the emptiness problem of deterministic propositional while programs with graph loop: Hypothesis elimination using loops, 2025. arXiv:2504.20415v1.

29 Peter W. O'Hearn. Incorrectness logic. *Proceedings of the ACM on Programming Languages*, 4(POPL):10:1–10:32, 2019. doi:10.1145/3371078.

30 Damien Pous. On the positive calculus of relations with transitive closure. In *Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 96 of *LIPIcs*, pages 3:1–3:16. Schloss Dagstuhl, 2018. doi:10.4230/LIPIcs.STACS.2018.3.

31 Damien Pous and Jana Wagemaker. Completeness theorems for Kleene algebra with tests and top. *Logical Methods in Computer Science*, Volume 20, Issue 3, 2024. doi:10.46298/lmcs-20(3:27)2024.

32 Jorge Pérez, Marcelo Arenas, and Claudio Gutierrez. nSPARQL: A navigational language for RDF. In *International Semantic Web Conference (ISWC)*, volume 5318 of *LNISA*, pages 66–81. Springer, 2008. doi:10.1007/978-3-540-88564-1_5.

33 Juan L. Reutter. Containment of nested regular expressions, 2013. arXiv:1304.2637v2.

34 Igor Sedlár. Kleene algebra with dynamic tests: Completeness and complexity, 2023. arXiv:2311.06937v1.

35 Igor Sedlár. On the complexity of Kleene algebra with domain. In *International Conference on Relational and Algebraic Methods in Computer Science (RAMiCS)*, volume 13896 of *LNCS*, pages 208–223. Springer, 2023. doi:10.1007/978-3-031-28083-2_13.

36 Alfred Tarski. On the calculus of relations. *The Journal of Symbolic Logic*, 6(3):73–89, 1941. doi:10.2307/2268577.

37 Ken Thompson. Programming techniques: Regular expression search algorithm. *Communications of the ACM*, 11(6):419–422, 1968. doi:10.1145/363347.363387.

38 Lena Verscht and Benjamin Lucien Kaminski. A taxonomy of Hoare-like logics: Towards a holistic view using predicate transformers and Kleene algebras with top and tests. *Proc. ACM Program. Lang.*, 9(POPL):60:1782–60:1811, 2025. doi:10.1145/3704896.

39 Cheng Zhang, Arthur Azevedo de Amorim, and Marco Gaboardi. Domain reasoning in TopKAT. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 297 of *LIPIcs*, pages 157:1–157:18. Schloss Dagstuhl, 2024. doi:10.4230/LIPIcs.ICALP.2024.157.