論文 / 著書情報
Article / Book Information

題目(和文)	
Title(English)	Computational Complexity of Several Extensions of Kleene Algebra
著者(和文)	中村誠希
Author(English)	Yoshiki Nakamura
出典(和文)	学位:博士(理学), 学位授与機関:東京工業大学, 報告番号:甲第11065号, 授与年月日:2019年3月26日, 学位の種別:課程博士, 審査員:鹿島 亮,南出 靖彦,伊東 利哉,田中 圭介,森 立平
Citation(English)	Degree:Doctor (Science), Conferring organization: Tokyo Institute of Technology, Report number:甲第11065号, Conferred date:2019/3/26, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

TOKYO INSTITUTE OF TECHNOLOGY

DOCTORAL THESIS

Computational Complexity of Several Extensions of Kleene Algebra

---

Author:
NAKAMURA Yoshiki

Supervisor:
Dr. KASHIMA Ryo

A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Science

in the

Mathematical and Computing Science

February 26, 2019

Acknowledgements

I am grateful to my supervisor, Ryo Kashima. He supported my research from my Bachelor degree in many aspects.

I would like to thank Yasuhiko Minamide, Toshiya Ito, Keisuke Tanaka, and Ryuhei Mori, who refereed this thesis and made remarks and beneficial comments to improve this thesis. I would also like to thank Ryoma Sin'ya, who also made remarks and beneficial comments to improve this thesis.

Contents

Chapter 1

Introduction

Kleene algebra [51] is an algebraic generalization of regular expressions, which consists of two binary operations, union ($\cup$) and concatenation ($\cdot$), two nullary operations, the identity of union (0) and the identity of concatenation (1), and one unary operation, iteration ($\bullet^*$). Iteration is a basic operation for expressing behaviors of programs or expressing properties of models. *Language semantics* and *relational semantics* are two of principal semantics in Kleene algebra. Each Kleene algebra term in language semantics is known as a regular expression. Relational semantics of Kleene algebra is related to relation algebra and logics (e.g., dynamic logic, description logic, temporal logic, first-order logic). Actually the two equational theories of Kleene algebra defined in language semantics and in relational semantics are equivalent. As for decidability, the theory (i.e., the equivalence problem for regular expressions) is decidable and PSPACE-complete [43]. This thesis "Computational Complexity of Several Extensions of Kleene Algebra" deals with decidability and computational complexity of several extensions of Kleene algebra.

The relational semantics of Kleene algebra is originated with *the calculus of (binary) relations* in the 1870s (according to the paper of Tarski [84]). The calculus of relations consists of two binary operations, union ($\cup$) and concatenation ($\cdot$), two nullary operations, the identity of union (0) and the identity of concatenation (1), and two unary operations, converse ($\bullet^\smile$) and complement ($\bullet^-$). The calculus of relations is equivalent to FO3 (the three variable fragment of first-order logic) in the sense of expressive power of binary relations [85] (see also [34]). Unfortunately, as for decidability, the

equational theory of the calculus of relations (and also the theory of FO3) is undecidable [85] and RE-complete. In connection with the undecidability result and with the undecidability of the validity problem for first-order logic (a.k.a. the Entscheidungsproblem) [24], many decidable and undecidable variants are studied (we refer to the book [16] for decidable fragments and undecidable fragments of first-order logic). The *positive* fragment [1] (called the positive calculus of relations) is one of decidable fragments of the calculus of relations, which is the calculus of relations without complement. The positive fragment is related to the *graph homomorphism problem*. Actually the decidability can be shown by reducing to finitely many graph homomorphism problems.

The positive calculus of relations extended with iteration in Kleene algebra is called *Kleene allegories* [18] (or called *the positive calculus of relations with transitive closure* [72]). In LICS '15, it was shown that the identity-free fragment of Kleene allegories (i.e., Kleene allegories without 1, $\bullet^\smile$, and $\bullet^*$ and with transitive closure) is decidable in EXPSPACE [18], but the decidability of the equational theory was open for Kleene allegories. In Chapter 3 we solve the problem positively. More precisely, we show that the equational theory of Kleene allegories is decidable and EXPSPACE-complete. The equational theory can be characterized by graph homomorphism problems like for the positive calculus of relations. However we cannot solve the equational theory by graph homomorphism problems because the number of the graph homomorphism problems reduced by an equational formula may be infinite. For that reason, we need an ingenious. For solving this problem, we introduce *derivatives on graphs* based on derivatives on strings for regular expressions introduced by Brzozowski [19] and Antimirov [3]. These derivatives reduce the equational theory of Kleene allegories to the equivalence problem for nondeterministic finite automata. The lower bound is proved by the EXPSPACE-hardness of the universality problem for regular expressions with intersection.

The language semantics of Kleene algebra is originated with Kleene's *regular events* (also called regular languages or regular sets) in 1950s [50]. The class of regular languages is one of well-studied language classes in formal language theory and is one of well-studied notions in computer science. Each term of Kleene algebra

in language semantics is called a *regular expression*. The equivalence problem for regular expressions is decidable and PSPACE-complete [43]. (The problem can be solved, for example, by reducing to the equivalence problem of nondeterministic finite automata [86].)

As described above, the equivalence problem for Kleene algebra terms under the normal equivalence is decidable, but the complexity class is PSPACE-hard. It is interesting to study whether there is a weak relation on Kleene algebra terms such that the equivalence problem under the relation is more effectively solvable than the one under the normal equivalence. As for first-order logic, it is known that we can obtain a decidable variant by weakening the validity condition as follows: the finite validity problem is undecidable (see e.g., [55, Thm. 9.2]), but the "almost surely" validity problem is decidable and PSPACE-complete [36] (we refer the book [55, Sec. 12]). In Chapter 4, we introduce a weak equivalence, called *p-equivalence*, as analogy of the notion of almost surely valid. In other words, the p-equivalence problem in language semantics is the almost surely validity problem for equational formulas of Kleene algebra over word (string) structures. (More formally, the p-equivalence is defined as follows: two languages are p-equivalent if and only if the limit of the density of the symmetric difference of them is 0.) While the almost surely validity problem is easier than the validity problem for first-order logic, the equivalence problem under p-equivalence (more strongly, any binary relation between p-equivalence and the normal equivalence) is *not easier* than the equivalence problem for regular expressions. For example, for regular expressions, we show that the p-equivalence problem is still PSPACE-hard (note that the normal equivalence problem is PSPACE-complete). This is shown by modifying the proof of the PSPACE-hardness of the universality problem for regular expressions. Moreover the emptiness problem under p-equivalence is also PSPACE-hard, whereas the emptiness problem under the normal equivalence is effectively solvable (more precisely, NC^1-complete). In a nutshell, the key of the modification is to consider *prefix-closed languages* (or to reduce to prefix-closed languages). For showing the upper bounds of the p-equivalence problem, we first characterize p-equivalence by *logical formulas over finite automaton structures* and then we give the upper bounds of the p-equivalence problem for several models of

regular languages (regular expressions, nondeterministic finite automata, and deterministic finite automata) by using the results in descriptive complexity theory (we refer to the book [44]). Finally, we also introduce p-equivalence for Kleene algebra in relational semantics and compare the two p-equivalences. In other words, the p-equivalence problem in relational semantics is the almost surely validity problem for equational formulas of Kleene algebra over structures of binary relations. The two p-equivalences are incomparable. We also show that, by using the fact that the zero-one law holds and the almost surely validity problem is decidable for first-order logic with least-fixed point operator [10], (1) every equational formula of Kleene algebra terms satisfies the zero-one law over structures of binary relations, whereas it is not the case over word structures (see [80] for the zero-one law of Kleene algebra terms over word structures); and (2) the almost surely equivalence problem for Kleene algebra terms over structures of binary relations is decidable.

Outline

This thesis consists of five chapters including the chapter of introduction and the chapter of concluding remarks. We outline the other three chapters as follows:

- Chapter 2 "Preliminaries": In this chapter we first give the definition of predicate logic, and then we give some fundamental results of Kleene algebra in relational semantics and language semantics. In the section for Kleene algebra in relation semantics, we show that it has a connection with relation algebra and first-order logic and we introduce fundamental results of relation algebra. In the section for Kleene algebra in language semantics, we introduce derivatives for regular expressions. Actually the two equational theories of Kleene algebra defined in relational semantics and language semantics are equivalent. Finally, we introduce an algebraic axiomatization of the equational theory.

- Chapter 3 "Derivatives for Kleene Allegories": In this chapter we show that the equational theory of Kleene algebra with relational intersection and relational converse in relational semantics (called *Kleene allegories* [18]) is decidable and EXPSPACE-complete. This problem was open in LICS '15. The equational

theory of Kleene allegories can be characterized by infinitely many graph homomorphism problems. For showing decidability, we introduce derivatives on graphs based on derivatives on strings for regular expressions introduced by Antimirov [3]. These derivatives reduce the equational theory of Kleene allegories to the equivalence problem for nondeterministic finite automata. The lower bound is proved by the EXPSPACE-hardness of the universality problem for regular expressions with intersection.

- Chapter 4 "Kleene Algebra Under Weak Equivalences": In this chapter we study the equivalence problem for Kleene algebra in language semantics (i.e., regular expressions) under weak equivalences. In particular, we introduce *p-equivalence* defined as follows: two languages are p-equivalent if and only if the limit of the density of the symmetric difference of them is 0. We give several logical characterizations for p-equivalence and give the complexity upper bounds by the results of descriptive complexity theory. For lower bound, we show that each language problem (e.g., the equivalence problem, the universality problem) under p-equivalence (more strongly, any binary relation between p-equivalence and the normal equivalence) is not easier than the one under normal equivalence in several cases. Finally, we introduce p-equivalence for Kleene algebra in relational semantics and compare the two p-equivalences.

Chapter 2

Preliminaries

2.1 General Notation

The symbol $\mathbb{N} = \{0, 1, \dots\}$ denotes the set of non-negative integers and the symbol $\mathbb{N}_+ = \{1, 2, \dots\}$ denotes the set of positive integers. $[n]$ denotes the interval $\{1, \dots, n\}$ for $n \in \mathbb{N}$.

The symbol $\emptyset$ (or $\{\}$) denotes the *empty set*. The *cardinality* of a set X is denoted by $\#(X)$. The *powerset* of a set X is denoted by $\wp(X)$. Let X_1 and X_2 be sets. The *symmetric difference* of X_1 and X_2, written $X_1 \triangle X_2$, is defined by $X_1 \triangle X_2 = (X_1 \setminus X_2) \cup (X_2 \setminus X_1)$. The *disjoint union* of X_1 and X_2, written $X_1 \uplus X_2$, is defined by $X_1 \uplus X_2 = \{(x_1, 1) \mid x_1 \in X_1\} \cup \{(x_2, 2) \mid x_2 \in X_2\}$. When i is clear from the context, (x, i) may be abbreviated as x. The *domain* and the *codomain* of a partial function $f : A \to B$ are denoted by $\mathrm{dom}(f)$ and $\mathrm{cod}(f)$, respectively.

Let A be an *alphabet* (a set of *characters*). A *string* over A is a finite sequence of characters. A^* denotes the set of all strings over A, and A^n denotes the set of all strings of length n over A. A *language* over A is a subset of A^*. We use $a \in A$ to denote a character, $s \in A^*$ to denote a string, and $\mathcal{L} \subseteq A^*$ to denote a language. $a_1 a_2 \dots a_n$ denotes the sequence $(a_1, a_2, \dots, a_n)$. The *empty string* ε is the empty sequence $()$. The *length* of a string s is denoted by $\|s\|$. The *concatenation* of strings $s = a_1 \dots a_n$ and $s' = a'_1 \dots a'_m$, written $s \cdot s'$, is the string $a_1 \dots a_n a'_1 \dots a'_m$. The symbol $\cdot$ may be omitted for short, i.e., ss' denotes $s \cdot s'$. The *n-th iterate of s*, written s^n, is inductively defined as follows: (1) $s^0 := \varepsilon$; (2) $s^{n+1} := s \cdot s^n$.

A *binary relation* (relation, for short) R on a set X is a subset of $X \times X$. The *identity relation* on a set X, written $\triangle(X)$, is defined by $\triangle(X) := \{(x, x) \mid x \in X\}$. The *composition* of R_1 and R_2, written $R_1 \cdot R_2$, is defined by $R_1 \cdot R_2 := \{(x, x'') \mid \exists x'.(x, x') \in R_1 \wedge (x', x'') \in R_2\}$. The *functional composition* of R_1 and R_2, written $R_1 \circ R_2$, is defined by $R_1 \circ R_2 := \{(x, x'') \mid \exists x'.(x, x') \in R_2 \wedge (x', x'') \in R_1\}$, i.e., $R_1 \circ R_2 := R_2 \cdot R_1$. The *relational converse* of R, written $R^\smile$, is defined by $R^\smile := \{(x', x) \mid (x, x') \in R\}$. The *n-th iterate of* R, written R^n, is inductively defined as follows: (1) $R^0 := \triangle(X)$; (2) $R^{n+1} := R \cdot R^n$. The *reflexive transitive closure* of R is denoted by R^*, i.e., $R^* := \bigcup_{n \geq 0} R^n$. The *transitive closure* of R is denoted by R^+, i.e., $R^+ := \bigcup_{n \geq 1} R^n$. $R(x)$ denotes the set $\{x' \in X \mid (x, x') \in R\}$.

2.2 First-order Logic and Second-order Logic

In this section we define the syntax and the semantics of first-order logic and second-order logic by reason of that some notions in this thesis are based on or related to logic.

2.2.1 Syntax

A *ranked alphabet* is an alphabet A with a function $\mathrm{ar} : A \to \mathbb{N}$, where $\mathrm{ar}(a)$ denotes the arity of a. Explicitly a^k denotes that the character a has arity k. A *signature* σ is a pair of two ranked alphabets, S^{F} and S^{R}, where S^{F} and S^{R} are disjoint. Every character in S^{F} is called a *function symbol* and every character in S^{R} is called a *relation symbol*. Every nullary function symbol is called a *constant symbol*. In particular, a signature with no relation symbols is called an *algebraic signature*.

Definition 2.2.1 (Term)**.** The set of *terms* over a signature σ and an alphabet A, written $\mathcal{T}_A^\sigma$, is the smallest set satisfying the follows: (1) $a \in \mathcal{T}_A^\sigma$ for $a \in A$; (2) if t_1 , t_2, $\ldots$, t_k are terms in $\mathcal{T}_A^\sigma$, then $f(t_1, t_2, \ldots, t_k)$ is also a term in $\mathcal{T}_A^\sigma$, for $f^k \in S_\sigma^{\mathsf{F}}$.

If f is a nullary function symbol, f denotes the term $f()$. If f is an unary function symbol, $(t)^f$ denotes the term $f(t)$. If f is a binary function symbol, $(t_1\ f\ t_2)$ denotes the term $f(t_1, t_2)$. We may denote a unary function symbol f by $\bullet^f$. We omit

parentheses and we use them in ambiguous situations when it is not clear how to parse.

Example 2.2.2. Let σ be the algebraic structure $(+^2, 0^0, \bullet^{-1})$, .i.e., $+$, 0, and -1 are all function symbols and have arity 2, 0, and 1, respectively. Let A be the alphabet $\{x\}$. Then $(x+0)+x^{-1}$ denotes the term $+(+(x,0),-1(x)) \in \mathcal{T}_A^\sigma$.

We now define formulas of first-order logic (FO) and second-order logic (SO).

Definition 2.2.3 (SO formula)**.** The set of SO *formulas* over a signature σ, an alphabet A, and a ranked alphabet $\mathcal{X}$, written $\Phi_{A,\mathcal{X}}^{\mathsf{SO},\sigma}$, is the smallest set satisfying the follows:

(1) If $R \in S^{\mathsf{R}}$ and $t_1, \ldots, t_k \in \mathcal{T}_A^\sigma$, then $R(t_1, \ldots, t_k) \in \Phi_{A,\mathcal{X}}^{\mathsf{SO},\sigma}$.

(2) If $t_1, t_2 \in \mathcal{T}_A^\sigma$, then $t_1 = t_2 \in \Phi_{A,\mathcal{X}}^{\mathsf{SO},\sigma}$.

(3) If $\varphi_1, \varphi_2 \in \Phi_{A,\mathcal{X}}^{\mathsf{SO},\sigma}$, then $\varphi_1 \wedge \varphi_2, \varphi_1 \vee \varphi_2, \neg\varphi_1 \in \Phi_{A,\mathcal{X}}^{\mathsf{SO},\sigma}$.

(4) If $x \in A$ and $\varphi \in \Phi_{A,\mathcal{X}}^{\mathsf{SO},\sigma}$, then $\exists x.\varphi, \forall x.\varphi \in \Phi_{A,\mathcal{X}}^{\mathsf{SO},\sigma}$.

(5) If $X^k \in \mathcal{X}$ and $t_1, \ldots, t_k \in \mathcal{T}_A^\sigma$, then $X(t_1, \ldots, t_k) \in \Phi_{A,\mathcal{X}}^{\mathsf{SO},\sigma}$.

(6) If $X \in \mathcal{X}$ and $\varphi \in \Phi_{A,\mathcal{X}}^{\mathsf{SO},\sigma}$, then $\exists X.\varphi, \forall X.\varphi \in \Phi_{A,\mathcal{X}}^{\mathsf{SO},\sigma}$.

Every $x \in A$ is called a *first-order variable* and every $X \in \mathcal{X}$ is called a *second-order variable*. The set of first-order free variables of a formula φ is denoted by $FV1(\varphi)$ and the set of second-order free variables of a formula φ is denoted by $FV2(\varphi)$. A formula φ is called a *sentence* if φ has no free variables, i.e., $FV1(\varphi) = FV2(\varphi) = \emptyset$.

We use $\mathbf{Q}$ to denote a quantifier, either $\exists$ or $\forall$. Then $\mathbf{Q}x$ is called a *first-order quantifier* and $\mathbf{Q}X$ is called a *second-order quantifier*. The following abbreviations are used: (1) $t_1 \neq t_2 :\equiv \neg(t_1 = t_2)$, (2) $\varphi_1 \to \varphi_2 :\equiv (\neg\varphi_1) \vee \varphi_2$, (3) $\varphi_1 \leftrightarrow \varphi_2 :\equiv (\varphi_1 \to \varphi_2) \wedge (\varphi_2 \to \varphi_1)$, (4) $\varphi_1 \nleftrightarrow \varphi_2 :\equiv \neg(\varphi_1 \leftrightarrow \varphi_2)$, (5) $(\forall x \mid \varphi).\psi :\equiv \forall x.(\varphi \to \psi)$, (6) $(\exists x \mid \varphi).\psi :\equiv \exists x.(\varphi \wedge \psi)$, (7) $(\forall X \mid \varphi).\psi :\equiv \forall X.(\varphi \to \psi)$, and (8) $(\exists X \mid \varphi).\psi :\equiv \exists X.(\varphi \wedge \psi)$.

Definition 2.2.4 (FO formula)**.** An SO formula φ is called an FO *formula* if φ contains no second-order variables and no second-order quantifiers.

Definition 2.2.5 (Equantional formula)**.** An SO formula $\varphi \in \Phi_{A,\mathcal{X}}^{\mathsf{SO},\sigma}$ is called an *equational formula* if φ is of the form $t_1 = t_2$, where $t_1, t_2 \in \mathcal{T}_A^\sigma$.

Definition 2.2.6 (MSO formula)**.** An SO formula φ is called a *k-adic SO formula* if every second-order variable occurring in φ has arity at most k. In particular, 1-adic SO is called *monadic* SO *(*MSO*)*.

Definition 2.2.7 (SO$\exists$ and SO$\forall$)**.** Let φ be an SO formula. Then φ is called an SO$\mathbf{Q}$ *formula* if φ is of the form $\mathbf{Q}X_1.\cdots\mathbf{Q}X_n.\psi$, where $n \geq 0$ and ψ contains no second-order quantifiers.

Definition 2.2.8 (FO$\exists$ and FO$\forall$)**.** Let φ be an FO formula. Then φ is called an FO$\mathbf{Q}$ *formula* if φ is of the form $\mathbf{Q}x_1.\cdots\mathbf{Q}x_n.\psi$, where $n \geq 0$ and ψ contains no first-order quantifiers (i.e., ψ is a quantifier-free formula).

We now define the size of terms and formulas, which is used as an induction measure or a complexity measure.

Definition 2.2.9 (Size)**.** The *size* of a term $t \in \mathcal{T}_A^\sigma$, written $\|t\|$, and the *size* of a formula $\varphi \in \Phi_{A,\mathcal{X}}^{\mathsf{SO},\sigma}$, written $\|\varphi\|$, are inductively defined as follows:

(1) $\|a\| := 1$ for $a \in A$;

(2) $\|f^k(t_1,\ldots,t_k)\| := \|X^k(t_1,\ldots,t_k)\| := \|R^k(t_1,\ldots,t_k)\| := 1 + \sum_{i=1}^k \|t_i\|$ for $f^k \in S^{\mathsf{F}}$, $X^k \in \mathcal{X}$, and $R^k \in S^{\mathsf{R}}$;

(3) $\|t_1 = t_2\| := 1 + \|t_1\| + \|t_2\|$;

(4) $\|\varphi_1 \wedge \varphi_2\| := \|\varphi_1 \vee \varphi_2\| := 1 + \|\varphi_1\| + \|\varphi_2\|$;

(5) $\|\neg\varphi\| := \|\exists x.\varphi\| := \|\forall x.\varphi\| := \|\exists X.\varphi\| := \|\forall X.\varphi\| := 1 + \|\varphi\|$.

2.2.2 Semantics

We now define the semantics of first-order logic and second-order logic.

Definition 2.2.10 (Structure)**.** A *structure* over a signature σ, written $\mathcal{A}$, is a tuple $(|\mathcal{A}|, \{f^{\mathcal{A}}\}_{f\in S_\sigma^{\mathsf{F}}}, \{R^{\mathcal{A}}\}_{R\in S_\sigma^{\mathsf{R}}})$, where

(1) $|\mathcal{A}|$ is a nonempty set,

(2) $f^{\mathcal{A}} : |\mathcal{A}|^{\mathrm{ar}(f)} \to |\mathcal{A}|$ is a function (if $\mathrm{ar}(f) = 0$, $f^{\mathcal{A}}()$ is an element in $|\mathcal{A}|$), and

(3) $R^{\mathcal{A}} \subseteq |\mathcal{A}|^{\mathrm{ar}(R)}$ is a relation.

In particular, a structure without relation symbols is called an *algebraic structure* (or called algebra).

Definition 2.2.11 (Model)**.** An *interpretation* (over A and $\mathcal{X}$) is a tuple $(\{x^M\}_{x\in A}, \{X^M\}_{X\in\mathcal{X}})$, where (1) x^M is an element of $|\mathcal{A}|$; and (2) $X^M \subseteq |\mathcal{A}|^{\mathrm{ar}(X)}$ is a relation. A *model* over A and $\mathcal{X}$, written M (or $(\mathcal{A}, I)$), is a structure $\mathcal{A}$ with an interpretation I. The *size* of a model M, written $\|M\|$, is defined as $\#(|M|)$.

Definition 2.2.12 (Semantics)**.** The *semantics of a term* $t \in \mathcal{T}_A^\sigma$ on a model M, written $[\![t]\!]_M$, is an element of $|M|$ inductively defined as follows:

(1) $[\![a]\!]_M := a^M$ for $a \in A$;

(2) $[\![f^k(t_1, \ldots, t_k)]\!]_M := f^M([\![t_1]\!]_M, \ldots, [\![t_k]\!]_M)$ for $f^k \in S_\sigma^{\mathsf{F}}$.

The *semantics of a formula* $\varphi \in \Phi_{A,\mathcal{X}}^{\mathsf{SO},\sigma}$ on a model M, written $[\![\varphi]\!]_M$, is a truth value (i.e., an element of $\{true, false\}$). $M \models \varphi$ (resp. $M \not\models \varphi$) denotes $[\![\varphi]\!]_M = true$ (resp. $[\![\varphi]\!]_M = false$). $M[a/x]$ denotes the model M in which x^M has been replaced by the element $a \in |M|$. $M[R/X]$ denotes the model M in which X^M has been replaced by the relation $R \subseteq |M|^{\mathrm{ar}(X)}$. Then $[\![\varphi]\!]_M$ is inductively defined as follows:

(1) $M \models X(t_1, \ldots, t_k) \ :\Longleftrightarrow\ ([\![t_1]\!]_M, \ldots, [\![t_k]\!]_M) \in X^M$ for $X^k \in \mathcal{X}$;

(2) $M \models R(t_1, \ldots, t_k) \ :\Longleftrightarrow\ ([\![t_1]\!]_M, \ldots, [\![t_k]\!]_M) \in R^M$ for $R^k \in S_\sigma^{\mathsf{R}}$;

(3) $M \models t_1 = t_2 \ :\Longleftrightarrow\ [\![t_1]\!]_M = [\![t_2]\!]_M$;

(4) $M \models \varphi_1 \vee \varphi_2 \ :\Longleftrightarrow\ M \models \varphi_1$ or $M \models \varphi_2$;

(5) $M \models \varphi_1 \wedge \varphi_2 \ :\Longleftrightarrow\ M \models \varphi_1$ and $M \models \varphi_2$;

(6) $M \models \neg\varphi \ :\Longleftrightarrow\ M \not\models \varphi$;

(7) $M \models \exists x.\varphi \ :\Longleftrightarrow$ there is $a \in |M|$ such that $M[a/x] \models \varphi$;

(8) $M \models \forall x.\varphi \ :\Longleftrightarrow$ for any $a \in |M|$, $M[a/x] \models \varphi$;

(9) $M \models \exists X^k.\varphi \ :\Longleftrightarrow$ there is $R \subseteq |M|^k$ such that $M[R/X] \models \varphi$;

(10) $M \models \forall X^k.\varphi \ :\Longleftrightarrow$ for any $R \subseteq |M|^k$, $M[R/X] \models \varphi$.

We also define the *semantics* of a formula φ on a structure $\mathcal{A}$ as follows: $[\![\varphi]\!]_{\mathcal{A}} = \{I \text{ is an interpretation} \mid (\mathcal{A}, I) \models \varphi\}$. $[\![\varphi]\!]_{\mathcal{A}}^{\vec{x}}$ denotes the set $\{(I(x_1), \ldots, I(x_{\|\vec{x}\|})) \in$

$|\mathcal{A}|^{\|\vec{x}\|} \mid I \in [\![\varphi]\!]_{\mathcal{A}}\}$, where $\vec{x}$ is a string over A and x_i denotes the i th character of $\vec{x}$. In the same manner $[\![\varphi]\!]^{\vec{X}}_{\mathcal{A}}$ denotes the set $\{(I(X_1), \ldots, I(X_{\|\vec{X}\|})) \in \wp(|\mathcal{A}|)^{\|\vec{X}\|} \mid I \in [\![\varphi]\!]_{\mathcal{A}}\}$, where $\vec{X}$ is a string over $\mathcal{X}$ and X_i denotes the i th character of $\vec{X}$. We call that a structure $\mathcal{A}$ satisfies a formula φ (denoted by $\mathcal{A} \models \varphi$) if $(\mathcal{A}, I) \models \varphi$ holds for any interpretation I. A formula φ is called a *theorem* if $\mathcal{A} \models \varphi$ holds for any structure $\mathcal{A}$.

We may extend formulas with a strict linear order ($<$) (see also "$<$" in [44]). The symbol $<$ is a special binary relation symbol (like the symbol $=$) interpreted as a strict linear order on $|M|$. Then we also extend formulas with the binary relation symbol $\leq$ and the constant symbols, $\overline{0}$ and $\overline{\max}$ to denote the non-strict linear order of $<$, the minimum element under $<$, the maximum element under $<$, respectively. $\mathsf{SO}^<$ (resp. $\mathsf{FO}^<, \ldots$) denotes the SO (resp. $\mathsf{FO}, \ldots$) with the following symbols: $<$, $\leq$, $\overline{0}$, and $\overline{\max}$. We may also extend formulas with transitive closure (TC) (see also "TC" in [44]). Then we assume that if φ is a formula, $(\mathrm{TC}_{x,x'}(\varphi))(y, y')$ is also a formula, where x, x', y, and y' are first-order variables, and $(\mathrm{TC}_{X,X'}(\varphi))(Y, Y')$ is also a formula, where X, X', Y, and Y' are second-order variables. The semantics of these formulas on $M = (\mathcal{A}, I)$ are defined as follows:

$$M \models (\mathrm{TC}_{x,x'}(\varphi))(y, y') \iff ([\![y]\!]_M, [\![y']\!]_M) \in ([\![\varphi]\!]^{x,x'}_{\mathcal{A}})^+$$

$$M \models (\mathrm{TC}_{X,X'}(\varphi))(Y, Y') \iff ([\![Y]\!]_M, [\![Y']\!]_M) \in ([\![\varphi]\!]^{X,X'}_{\mathcal{A}})^+$$

where R^+ denotes the transitive closure of a binary relation R. We may also extend formulas with deterministic transitive closure (DTC) (see also "DTC" in [44]). The semantics on $M = (\mathcal{A}, I)$ of $\mathrm{DTC}_{x,x'}(\varphi))(y, y')$ is defined as follows:

$$M \models (\mathrm{DTC}_{x,x'}(\varphi))(y, y') \iff \begin{cases} M \models (\mathrm{TC}_{x,x'}(\varphi))(y, y') & ([\![\varphi]\!]^{x,x'}_{\mathcal{A}} \text{ is deterministic}) \\ \mathit{false} & (\mathit{otherwise}) \end{cases}$$

where A binary relation R on X is called *deterministic* if $\#\{x' \in X \mid (x, x') \in R\} = 1$ holds for any $x \in X$. $\mathsf{SO}(\mathrm{TC})$ (resp. $\mathsf{FO}^<(\mathrm{TC}), \ldots$) denotes the SO (resp. $\mathsf{FO}^<, \ldots$) with transitive closure. $\mathsf{SO}(\mathrm{DTC})$ (resp. $\mathsf{FO}^<(\mathrm{DTC}), \ldots$) denotes the SO (resp. $\mathsf{FO}^<$, $\ldots$) with deterministic transitive closure.

2.3 Kleene Algebra and Relation Algebra

In this section we introduce some fundamental results related to Kleene algebra or relation algebra. We introduce two semantics for Kleene algebra, *language semantics* and *relational semantics*. The relational semantics of Kleene algebra is also deeply related with relation algebra. Let σ^{KA} be the algebraic signature $(0^0, 1^0, \cdot^2, \cup^2, \bullet^*)$. A *term of Kleene algebra* (also called *regular expression*) is a term over the signature σ^{KA}. $\mathcal{T}_A^{\mathsf{KA}}$ denotes the set of all Kleene algebra terms over an alphabet A. $\Phi_A^{=,\mathsf{KA}}$ denotes the set of all equational formulas over σ^{KA} and an alphabet A. Roughly speaking, the symbol 0 denotes the identity element for the operator $\cup$, the symbol 1 denotes the identity element for the operator $\cdot$, the symbol $\cdot$ denotes concatenation, the symbol $\cup$ denotes union, and the symbol $\bullet^*$ (called Kleene star) denotes iteration. $t_1 t_2$ is an abbreviated for $t_1 \cdot t_2$. t^n denotes the n-th iterate of t, i.e., t^n is inductively defined as follows: (1) $t^0 :\equiv 1$; (2) $t^{n+1} :\equiv t \cdot t^n$.

2.3.1 Relational Semantics

The relational semantics of Kleene algebra is originated with *the calculus of (binary) relations* in the 1870s (according to the papar of Tarski [84]). A *binary relation* (relation, for short) R on a set X is a subset of $X \times X$. First we define some operations on binary relations. The *identity relation* on a set X is defined by $\triangle(X) := \{(x, x) \mid x \in X\}$. The *composition* of R_1 and R_2, written $R_1 \cdot R_2$, is defined by $R_1 \cdot R_2 := \{(x, x'') \mid \exists x'.(x, x') \in R_1 \wedge (x', x'') \in R_2\}$. The *functional composition* of R_1 and R_2, written $R_1 \circ R_2$, is defined by $R_1 \circ R_2 := \{(x, x'') \mid \exists x'.(x, x') \in R_2 \wedge (x', x'') \in R_1\}$, i.e., $R_1 \circ R_2 := R_2 \cdot R_1$. The *relational converse* of R, written $R^\smile$, is defined by $R^\smile := \{(x', x) \mid (x, x') \in R\}$. The *n-th iterate of* R, written R^n, is inductively defined as follows: (1) $R^0 := \triangle(X)$; (2) $R^{n+1} := R \cdot R^n$. The *reflexive transitive closure* of R is denoted by R^*, i.e., $R^* := \bigcup_{n \geq 0} R^n$. The *transitive closure* of R is denoted by R^+, i.e., $R^+ := \bigcup_{n \geq 1} R^n$.

Relational semantics is defined on the following model.

Definition 2.3.1 (Relational model)**.** A relational model M (over an alphabet A) is a tuple $(V, \{R_a\}_{a \in A})$, where V is a set and every $R_a \subseteq V^2$ is a binary relation on V.

The class of all relational models is denoted by REL_A. REL_A^n denotes the class of all relational models of size n, i.e., $REL_A^n = \{M \in REL_A \mid \#V_M = n\}$.

Definition 2.3.2 (Relational semantics)**.** The *relational semantics* of a term t in $\mathcal{T}_A^{\mathsf{KA}}$ on a relational model $M = (V, \{R_a\}_{a\in A})$, written $[\![t]\!]_M$, is a binary relation on V_M, inductively defined as follows:

(1) $[\![a]\!]_M := R_a$ for $a \in A$;

(2) $[\![0]\!]_M := \emptyset$;

(3) $[\![1]\!]_M := \triangle(V)$;

(4) $[\![t_1 \cdot t_2]\!]_M := [\![t_1]\!]_M \cdot [\![t_2]\!]_M$;

(5) $[\![t_1 \cup t_2]\!]_M := [\![t_1]\!]_M \cup [\![t_2]\!]_M$;

(6) $[\![t^*]\!]_M := [\![t]\!]_M^*$.

Remark. In fact every relational model M is also a model in Definition 2.2.11 (i.e., an algebraic structure over σ^{KA} on $\wp(V_M^2)$ with an interpretation).

The equational theory over relational semantics, written $\mathbf{RKA}_A$, is a set of equational formulas, defined by $\mathbf{RKA}_A := \{t_1 = t_2 \in \Phi_A^{=,\mathsf{KA}} \mid [\![t_1]\!]_M = [\![t_2]\!]_M \text{ for any } M \in REL_A\}$.

Proper Relation Algebra

The relational semantics for Kleene algebra (Definition 2.3.2) is based on the calculus of (binary) relations (and relation algebra) [84]. Let $\sigma^{\mathsf{RA}} = (0^0, 1^0, \top^0, \cdot^2, \bullet^\smile, \cup^2, \cap^2, \bullet^-)$ be the algebraic signature of relation algebra. $\mathcal{T}_A^{\mathsf{RA}}$ denotes the set of all terms over σ^{RA} and an alphabet A. $\Phi_A^{=,\mathsf{RA}}$ denotes the set of all equational formulas over σ^{RA} and an alphabet A. Roughly speaking, the symbols 0, 1, and $\top$ denote the identity element for the operators $\cup$, $\cdot$, and $\cap$, respectively, the symbol $\cdot$ denotes concatenation, the symbol $\bullet^\smile$ denotes the relational converse, and the symbols $\cup$, $\cap$, and $\bullet^-$ denote union, intersection, and complement in set theory. The following is an algebraic definition of the calculus of relations.

Definition 2.3.3 (Proper relation agebra, see e.g., [62, p.24.])**.** An algebraic structure $\mathcal{A}$ over σ^{RA} is called a *proper relation algebra* if the following hold:

(1) V is a set;

(2) $|\mathcal{A}| \subseteq \wp(V^2)$;

(3) $0^{\mathcal{A}} = \emptyset$;

(4) $1^{\mathcal{A}} = \triangle(V)$;

(5) $\top^{\mathcal{A}} = V^2$;

(6) $R_1 \cdot^{\mathcal{A}} R_2 = R_1 \cdot R_2$;

(7) $R^{\smile \mathcal{A}} = R^{\smile}$;

(8) $R_1 \cup^{\mathcal{A}} R_2 = R_1 \cup R_2$;

(9) $R_1 \cap^{\mathcal{A}} R_2 = R_1 \cap R_2$;

(10) $R^{-\mathcal{A}} = V^2 \setminus R$;

(11) $|\mathcal{A}|$ is closed under $\cdot$, $\bullet^{\smile}$, $\cup$, $\cap$, and complement (i.e., $V^2 \setminus R$), and $0^{\mathcal{A}}, 1^{\mathcal{A}}, \top^{\mathcal{A}} \in |\mathcal{A}|$,

where each R, R_1, and R_2 is any binary relation on V.

Remark. We omit the symbol "†" (the dual of $\cdot$) in [84] for considering positive fragments of the calculus of relations. However this restriction is not essential for the (full) calculus of relations because $R_1 \dagger R_2$ can be expressed as $(R_1^- \cdot R_2^-)^-$ by using other symbols. In the same manner we can reduce the signature σ^{RA} to $(1, \cup, \cdot, \bullet^{\smile}, \bullet^-)$ by the following equations: $\top = 1 \cup 1^-$; $0 = (1 \cup 1^-)^-$; and $R_1 \cap R_2 = (R_1^- \cup R_2^-)^-$.

An algebraic structure $\mathcal{A}$ over σ^{RA} is called a *representable relation algebra* [62, p.24] if $\mathcal{A}$ is isomorphic to a proper relation algebra. We denote the class of representable relation algebras by RRA. We are also interested in the following equational theory: $\mathbf{RRA}_A := \{t_1 = t_2 \in \Phi_A^{=,RA} \mid \mathcal{A} \models t_1 = t_2 \text{ for any } \mathcal{A} \in RRA\}$.

Remark. Every proper relation algebra (with an interpretation) is very closed to a relational model defined in the previous section. The *relational semantics* of a term t in $\mathcal{T}_A^{\mathsf{RA}}$ on a relational model $M = (V, \{R_a\})$, written $[\![t]\!]_M$, is defined like Definition 2.3.2, as follows:

(1) $[\![a]\!]_M = R_a$ for $a \in A$;

(2) $[\![0]\!]_M = \emptyset$;

(3) $[\![1]\!]_M = \triangle(V)$;

(4) $[\![\top]\!]_M = V^2$;

(5) $[\![t_1 \cdot t_2]\!]_M = [\![t_1]\!]_M \cdot [\![t_2]\!]_M$;

(6) $[\![t^\smile]\!]_M = [\![t]\!]_M^\smile$;

(7) $[\![t_1 \cup t_2]\!]_M = [\![t_1]\!]_M \cup [\![t_2]\!]_M$;

(8) $[\![t_1 \cap t_2]\!]_M = [\![t_1]\!]_M \cap [\![t_2]\!]_M$;

(9) $[\![t^-]\!]_M = V^2 \setminus [\![t]\!]_M$.

Actually the equational theory $\mathbf{RRA}_A$ can be also defined as $\{t_1 = t_2 \in \Phi_A^{=,RA} \mid [\![t_1]\!]_M = [\![t_2]\!]_M$ for any relational model $M\}$.

Relational Semantics and FO3

Let A be an alphabet, let $\sigma^{\mathcal{T}_A^{\mathsf{RA}}}$ be the signature $(\{t^2\}_{t \in \mathcal{T}_A^{\mathsf{RA}}})$, and let σ^A be the signature $(\{a^2\}_{a \in A})$, where every t (resp. a) is a binary relation symbol. The following (Figure 2.1) is an axiomatization of the calculus of relations, given by Tarski [84, p.75-76] (where the axioms for the symbol 1 is replaced from [84, p.75-76] by using the symbol $=$).

(R1) $\forall x.\forall y.((x \top y) \leftrightarrow (x = x))$
(R2) $\forall x.\forall y.((x\, 0\, y) \leftrightarrow \neg(x = x))$
(R3) $\forall x.\forall y.((x\, 1\, y) \leftrightarrow (x = y))$
(R4) $\forall x.\forall y.((x\, t^-\, y) \leftrightarrow \neg(x\, t\, y))$
(R5) $\forall x.\forall y.((x\, t^\smile\, y) \leftrightarrow (y\, t\, x))$
(R6) $\forall x.\forall y.((x\, (t_1 \cup t_2)\, y) \leftrightarrow ((x\, t_1\, y) \vee (x\, t_2\, y)))$
(R7) $\forall x.\forall y.((x\, (t_1 \cap t_2)\, y) \leftrightarrow ((x\, t_1\, y) \wedge (x\, t_2\, y)))$
(R8) $\forall x.\forall y.((x\, (t_1 \cdot t_2)\, y) \leftrightarrow (\exists z.((x\, t_1\, z) \wedge (z\, t_2\, y))))$

FIGURE 2.1: An axiomatization of the calculus of relations [84, p.75-76]

We denote the class of structures over $\sigma^{\mathcal{T}_A^{\mathsf{RA}}}$ satisfying (R1)-(R8) by $\mathcal{RS}$. Every structure in $\mathcal{RS}$ is essentially equivalent to a model of proper relation algebras.

Proposition 2.3.4. *Let $(\mathcal{A}, I)$ be any pair of a proper relation algebra and an interpretation, and let $\mathcal{A}'$ be any structure in $\mathcal{RS}$. If $[\![a]\!]_{(\mathcal{A},I)} = a^{\mathcal{A}'}$ holds for any $a \in A$, then*

(1) *for any term* $t \in \mathcal{T}_A^{\mathsf{RA}}$, $[\![t]\!]_{(\mathcal{A},I)} = t^{\mathcal{A}'}$;

(2) *for any terms* $t_1, t_2 \in \mathcal{T}_A^{\mathsf{RA}}$, $(\mathcal{A}, I) \models t_1 = t_2 \iff \mathcal{A}' \models t_1 = t_2$.

Proof Sketch. (1) is easily proved by induction on the structure of t. (2) is by (1). □

From this, $\mathbf{RRA}_A = \{t_1 = t_2 \in \Phi_A^{=,\mathsf{RA}} \mid \mathcal{A} \models t_1 = t_2 \text{ for any } \mathcal{A} \in \mathcal{RS}\}$ also holds. (Note that, for any model $(\mathcal{A}, I)$ of proper relation algebras, there is a structure $\mathcal{A}'$ in $\mathcal{RS}$ such that $[\![a]\!]_{(\mathcal{A},I)} = a^{\mathcal{A}'}$ holds for any $a \in A$, and vice versa.) Actually every term of relation algebra can be replaced by an FO3 formula by using (R1)-(R8). An FO formula is called an FO3 (resp. FOk) formula if the number of variables occurring is at most 3 (resp. k, where k is a natural number). For example, $\forall x.\forall y.\forall z.\forall x.x = x$ is an FO3 formula, but $\forall x.\forall y.\forall z.\forall w.x = x$ is not. Figure 2.2 gives a translation from every term of relation algebra to an FO3 formula over the signature σ^A and a set of first-order variables of size 3 (denoted by V_3).

(R′1) $\varphi_a(x,y) :\equiv (x\ a\ y)$
(R′2) $\varphi_\top(x,y) :\equiv (x = x)$
(R′3) $\varphi_0(x,y) :\equiv \neg(x = x)$
(R′4) $\varphi_1(x,y) :\equiv (x = y)$
(R′5) $\varphi_{t^-}(x,y) :\equiv \neg\varphi_t(x,y)$
(R′6) $\varphi_{t^\smile}(x,y) :\equiv \varphi_t(y,x)$
(R′7) $\varphi_{t_1 \cup t_2}(x,y) :\equiv \varphi_{t_1}(x,y) \vee \varphi_{t_2}(x,y)$
(R′8) $\varphi_{t_1 \cap t_2}(x,y) :\equiv \varphi_{t_1}(x,y) \wedge \varphi_{t_2}(x,y)$
(R′9) $\varphi_{t_1 \cdot t_2}(x,y) :\equiv \exists z.(\varphi_{t_1}(x,z) \wedge \varphi_{t_2}(z,y))$
where $x, y \in V_3$ and z is a variable in V_3 such that z is not x and is not y.

FIGURE 2.2: From relation algebra terms to FO3 formulas [34, Sec. 8.]

Theorem 2.3.5 (e.g., [34, Sec. 8]). *Let u and v be any variables. Then*

(1) *For any term t in $\mathcal{T}_A^{\mathsf{RA}}$, there is an* FO3 *formula φ' over σ^A such that $t^{\mathcal{A}} = [\![\varphi']\!]_{\mathcal{A}}^{u,v}$ holds for any $\mathcal{A} \in \mathcal{RS}$.*

(2) *For any* FO3 *formula φ over $\sigma^{\mathcal{T}_A^{\mathsf{RA}}}$, there is an* FO3 *formula φ' over σ^A such that $[\![\varphi]\!]_{\mathcal{A}}^{u,v} = [\![\varphi']\!]_{\mathcal{A}}^{u,v}$ holds for any structure $\mathcal{A}$ over σ^A.*

Proof Sketch. (1): $[\![t]\!]_M = [\![\varphi_t(u,v)]\!]_M^{u,v}$ is shown by induction on the structure of t. (2): This part is shown by (1). □

Actually the converse of Theorem 2.3.5 (1) also holds. To prove it, we define a class of formulas, *extended disjunctive normal form* (extended DNF). We call that an FO formula is an extended DNF if the formula is in the set Φ, which is defined by the following grammar:

$$\begin{aligned} \varphi \in \Phi &::= \varphi \vee \varphi \mid \psi \\ \psi &::= \psi \wedge \psi \mid \rho \mid \neg\rho \\ \rho &::= (x = y) \mid (x\ a\ y) \mid \exists x.\psi \end{aligned}$$

The only difference from the DNF is that it allows every formula of the form $\exists x.\psi$ as a literal. We call that a formula of the form ρ or $\neg\rho$ is an *extended literal* (denoted by $l, l', \ldots$). Every extended DNF can be expressed as $\bigvee_{i\in[n]} \bigwedge_{j\in[m_i]} l_{i,j}$. The *complement* of an extended literal l, written $\bar{l}$, is defined by $\bar{l} :\equiv \begin{cases} \rho & (l \equiv \neg\rho) \\ \neg\rho & (l \equiv \rho) \end{cases}$. Figure 2.3 gives a translation from every FO formula to an equivalent extended DNF FO formula.

(1) $(x = y)^T :\equiv (x = y)$
(2) $(x\ a\ y)^T :\equiv (x\ a\ y)$
(3) $(\varphi_1 \vee \varphi_2)^T :\equiv \varphi_1^T \vee \varphi_2^T$
(4) $(\varphi_1 \wedge \varphi_2)^T :\equiv (\neg(\neg\varphi_1 \vee \neg\varphi_2))^T$
(5) $(\neg\varphi)^T :\equiv \bigvee_{(j_1,\ldots,j_n)\in[m_1]\times\cdots\times[m_n]} \bigwedge_{i\in[n]} \bar{l}_{i,j_i}$, where $\varphi^T \equiv \bigvee_{i\in[n]} \bigwedge_{j\in[m_i]} l_{i,j}$,
(6) $(\exists x.\varphi)^T :\equiv \bigvee_{i\in[n]} (\exists x. \bigwedge_{j\in[m_i]} l_{i,j})$, where $\varphi^T \equiv \bigvee_{i\in[n]} \bigwedge_{j\in[m_i]} l_{i,j}$.
(7) $(\forall x.\varphi)^T :\equiv (\neg\exists x.\neg\varphi)^T$

FIGURE 2.3: From FO formulas to extended DNF formulas

Lemma 2.3.6. *Every* FO *(resp.* FO3*) formula φ is equivalent to an extended DNF* FO *(resp.* FO3*) formula.*

Proof Sketch. For any structure $\mathcal{A} \in \mathcal{RS}$ and any interpretation I, $[\![\varphi^T]\!]_{\mathcal{A},I} = [\![\varphi]\!]_{\mathcal{A},I}$ holds. It is easily proved by induction on the structure of φ, where we assume that $\varphi_1 \wedge \varphi_2$ is the abbreviation of $\neg(\neg\varphi_1 \vee \neg\varphi_2)$ and $\forall x.\varphi$ is the abbreviation of $\neg\exists x.\neg\varphi$. □

Explicitly $\rho^{\{u,v\}}$ (resp. $\psi^{\{u,v\}}$) denotes that ρ (resp. ψ) is a formula such that $FV1(\rho) \subseteq \{u, v\}$ (resp. $FV1(\psi) \subseteq \{u, v\}$). Note that the number of free variables of every ρ is at most 2, and thus every ψ can be expressed as $\psi^{\{x,y\}} \wedge \psi^{\{y,z\}} \wedge \psi^{\{z,x\}}$, where variables

x, y, and z are *distinct*. For that reason, we can redefine the definition of *extended DNF* as the set Φ defined by the following grammar (where x, y, z are all distinct):

$$\begin{aligned}
\varphi \in \Phi &::= \varphi \vee \varphi \mid \psi \\
\psi &::= \hat{\psi}^{\{x,y\}} \wedge \hat{\psi}^{\{y,z\}} \wedge \hat{\psi}^{\{z,x\}} \\
\hat{\psi}^{\{x,y\}} &::= \hat{\psi}^{\{x,y\}} \wedge \hat{\psi}^{\{x,y\}} \mid \rho^{\{x,y\}} \mid \neg\rho^{\{x,y\}} \\
\rho^{\{x,y\}} &::= (x = x) \mid (x = y) \mid (x\, a\, x) \mid (x\, a\, y) \mid \exists z.\psi
\end{aligned}$$

Remark. The above extended DNF is the key to translate to a term of relation algebra. Note that $[\![\varphi_1 \wedge \varphi_2]\!]^{u,v}_{\mathcal{A}} = [\![\varphi_1]\!]^{u,v}_{\mathcal{A}} \cap [\![\varphi_2]\!]^{u,v}_{\mathcal{A}}$ does not always hold. For example, if $[\![\varphi_1]\!]^{u,v,w}_{\mathcal{A}} = \{(0,0,0)\}$ and $[\![\varphi_2]\!]^{u,v,w}_{\mathcal{A}} = \{(0,0,1)\}$, then $[\![\varphi_1 \wedge \varphi_2]\!]^{u,v}_{\mathcal{A}} = \emptyset$, but $[\![\varphi_1]\!]^{u,v}_{\mathcal{A}} \cap [\![\varphi_2]\!]^{u,v}_{\mathcal{A}} = \{(0,0)\}$.

Figure 2.4 shows a translation from every extended DNF FO3 formula over σ^A to a term of relation algebra. By the translation, the following is shown.

Theorem 2.3.7 (e.g., [34, Sec. 20], [85, Sec. 3.9])**.** *Let u and v be any variables, and let φ be any* FO3 *formula over σ^A. Then there is a term $t \in \mathcal{T}^{\mathsf{RA}}_A$ such that $[\![\varphi]\!]^{u,v}_{\mathcal{A}} = t^{\mathcal{A}}$ holds for any $\mathcal{A} \in \mathcal{RS}$.*

Proof. See Appendix A. □

(1) $T_{u,u}(\varphi) :\equiv (T_{u,v}(\varphi) \cdot \top) \cap 1$
(2) $T_{u,v}(x = x) :\equiv \top$
(3) $T_{u,v}(x = y) :\equiv 1_{u\neq z} \cdot 1 \cdot 1_{v\neq z}$
(4) $T_{u,v}(x\, a\, x) :\equiv 1_{u=x} \cdot (a \cap 1) \cdot 1_{v=x}$
(5) $T_{u,v}(x\, a\, y) :\equiv \begin{cases} 1_{u\neq z} \cdot a \cdot 1_{v\neq z} & (u = x \vee v = y) \\ 1_{u\neq z} \cdot a^{\smile} \cdot 1_{v\neq z} & (u = y \vee v = x) \end{cases}$
(6) $T_{u,v}(\exists z.\psi) :\equiv \begin{cases} T_{u,z}(\psi) \cdot T_{z,v}(\psi) & (u \neq z \wedge v \neq z) \\ 1_{u\neq z} \cdot T_{u,v}(\psi) \cdot 1_{v\neq z} & (otherwise) \end{cases}$
(7) $T_{u,v}(\neg\rho^{\{x,y\}}) :\equiv 1_{u\neq z} \cdot T_{u,v}(\rho^{\{x,y\}})^{-} \cdot 1_{v\neq z}$
(8) $T_{x,y}(\hat{\psi}^{\{x,y\}} \wedge \hat{\psi}^{\{y,z\}} \wedge \hat{\psi}^{\{z,x\}}) :\equiv T_{x,y}(\hat{\psi}^{\{x,y\}}) \cap (T_{x,z}(\hat{\psi}^{\{z,x\}}) \cdot T_{z,y}(\hat{\psi}^{\{y,z\}}))$
(9) $T_{x,y}(\hat{\psi}_1^{\{x,y\}} \wedge \hat{\psi}_2^{\{x,y\}}) :\equiv T_{x,y}(\hat{\psi}_1^{\{x,y\}}) \cap T_{x,y}(\hat{\psi}_2^{\{x,y\}})$
(10) $T_{u,v}(\varphi_1 \vee \varphi_2) :\equiv T_{u,v}(\varphi_1) \cup T_{u,v}(\varphi_2)$

where x, y, and z are all distinct, u and v are distinct, and 1_φ denotes the term

$$\begin{cases} 1 & (\varphi \text{ holds}) \\ \top & (\text{otherwise}) \end{cases}.$$

FIGURE 2.4: From extended DNF FO3 formulas to relation algebra terms

By Theorem 2.3.5(1) and 2.3.7, FO3 formulas and relation algebra terms are equivalent in the sense of expressive power of binary relations.

Corollary 2.3.8. *Let t_1 and t_2 be any terms in $\mathcal{T}_A^{\mathsf{RA}}$ and let φ be any* FO3 *sentence over σ^A.*

(1) $t_1 = t_2 \in \mathbf{RRA}_A \iff \forall x.\forall y.\varphi_{t_1}(x,y) \leftrightarrow \varphi_{t_2}(x,y)$ *is a theorem.*

(2) φ *is a theorem* $\iff T_{x,y}(\varphi^T) = \top \in \mathbf{RRA}_A$, *where x and y are any two variables.*

If we assume that $\#A$ is infinite, the theory of FO3 over σ^A is RE-complete as a corollary to the undecidability of the $\forall\exists\forall$ case with only binary relation symbols [47] (see also [16, Thm. 3.1.1]), and thus $\mathbf{RRA}_A$ is also RE-complete. The upperbound is shown by that the theory of first-order logic is recursively enumerable by Gödel's completeness theorem.) More strongly, the RE-hardness holds even if $\#A \geq 1$.

Theorem 2.3.9 (cf. [63, p.399])**.** *(1)* $\mathbf{RRA}_A$ *is* RE*-complete for* $\#A \geq 1$.

(2) The theory of FO3 *formulas over σ^A with $=$ is* RE*-complete for* $\#A \geq 1$.

(3) The theory of FO3 *formulas over σ^A without $=$ is* RE*-complete for* $\#A \geq 2$.

Proof Sketch. (1): By [63, p.399]. (2): By (1) and Corollary 2.3.8(1). (3): By (2) and that $=$ can be simulated by adding a binary relation denoting an equivalence relation like [84, p.75-76]. □

Remark. In connection with Theorem 2.3.9(3), the theory of FO formulas with just one binary relation symbol and without $=$ is also RE-complete by the Church-Herbrand theorem (see e.g., [15, Thm. 21.4]). However, to the best of our knowledge, it is open whether the theory of FO3 formulas with just one binary relation symbol and without $=$ (or the equational theory of relation algebra terms with just one variable and without 1) is RE-complete. [1]

Now we consider a decidable fragment of relation algebra terms, called *positive relation algebra* terms. Let t be a relation algebra term. We call that t is *positive* if t does not contain $\bullet^-$ and t is *strictly positive* if t does not contain $\bullet^-$ and does not contain 0. We remark that strictly positive relation algebra terms and (strictly) *positive* FO3$\exists$ formulas are also equivalent in the sense of expressive power of binary relations. We call that a formula φ is *positive* if φ does not contain $\neg$ and that a formula φ is

[1] Actually in [71] we have proved that the theory of FO3 formulas with just one binary relation symbol and without $=$ is RE-complete.

positive FO3∃ formula if φ is positive and an FO3 formula and an FO∃ formula. The following is immediate from the translations in Figure 2.2, 2.3, and 2.4.

Corollary 2.3.10. *Let u and v be any variables. Then*

(1) For any strictly positive relation algebra term t, there is a positive FO3∃ *formula φ over σ^A such that $t^{\mathcal{A}} = [\![\varphi]\!]^{u,v}_{\mathcal{A}}$ holds for any $\mathcal{A} \in \mathcal{RS}$.*

(2) For any positive FO3∃ *formula φ over σ^A, there is a strictly positive relation algebra term t such that $[\![\varphi]\!]^{u,v}_{\mathcal{A}} = t^{\mathcal{A}}$ holds for any $\mathcal{A} \in \mathcal{RS}$.*

Actually the problem to decide whether a given equational formula is a theorem of (representable) positive relation algebra can be reduced to finitely many graph homomorphism problems (see e.g., [72, Thm. 16]) by that the graph language of every positive relation algebra is finite (see e.g., [72, Def. 15]). In Chapter 3, we consider about the equational theory of positive relation algebra with Kleene star (but without $\top$). Concerning Kleene star, the following also holds.

Corollary 2.3.11. *Let u and v be any variables. Then*

(1) For any term t of relation algebra with Kleene star, there is an FO3(TC) *formula φ over σ^A such that $t^{\mathcal{A}} = [\![\varphi]\!]^{u,v}_{\mathcal{A}}$ holds for any $\mathcal{A} \in \mathcal{RS}$.*

(2) For any FO3(TC) *formula φ over σ^A, there is a term t of relation algebra with Kleene star such that $[\![\varphi]\!]^{u,v}_{\mathcal{A}} = t^{\mathcal{A}}$ holds for any $\mathcal{A} \in \mathcal{RS}$.*

(3) For any term t of strictly positive relation algebra with Kleene star, there is a positive FO3∃(TC) *formula φ over σ^A such that $t^{\mathcal{A}} = [\![\varphi]\!]^{u,v}_{\mathcal{A}}$ holds for any $\mathcal{A} \in \mathcal{RS}$.*

(4) For any positive FO3∃(TC) *formula φ over σ^A, there is a term t of strictly positive relation algebra with Kleene star such that $[\![\varphi]\!]^{u,v}_{\mathcal{A}} = t^{\mathcal{A}}$ holds for any $\mathcal{A} \in \mathcal{RS}$.*

Proof Sketch. Note that $[\![x\ \mathrm{TC}_{u,v}(\varphi)\ y]\!]^{x,y,z} = ([\![\varphi]\!]^{u,v}_{\mathcal{A}})^+ \times |\mathcal{A}|$, where R^+ denotes the transitive closure of a binary relation R. We add the following rules to the translations in Figure 2.2, 2.3, and 2.4, respectively.

$$\varphi_{t^*}(u,v) :\equiv u = v \vee (u\ \mathrm{TC}_{u,v}(\varphi_t(u,v))\ v)$$

$$(u\ \mathrm{TC}_{s,t}(\varphi)\ v)^T :\equiv (u\ \mathrm{TC}_{s,t}(\varphi^T)\ v)$$

$$T_{u,v}(x\,\mathrm{TC}_{s,t}(\varphi)\,x) :\equiv 1_{u=x}\cdot(T_{s,t}(\varphi)^+\cap 1)\cdot 1_{v=x}$$

$$T_{u,v}(x\,\mathrm{TC}_{s,t}(\varphi)\,y) :\equiv \begin{cases} 1_{u\neq z}\cdot T_{s,t}(\varphi)^+\cdot 1_{v\neq z} & (u=x\vee v=y)\\ 1_{u\neq z}\cdot (T_{s,t}(\varphi)^+)^\smile\cdot 1_{v\neq z} & (u=y\vee v=x)\end{cases}$$

where t^+ denotes the term $t\cdot t^*$ and we regard formulas of the form $\mathrm{TC}_{x,y}(\varphi)$ as a literal in the translation in Figure 2.3. Then $[\![t]\!]_{\mathcal{A}} = [\![\varphi_t(u,v)]\!]^{u,v}_{\mathcal{A}}$ and $[\![\varphi]\!]^{u,v}_{\mathcal{A}} = [\![T_{u,v}(\varphi^T)]\!]_{\mathcal{A}}$ are proved in the same way as the above. □

Remark. The translation in Figure 2.4 does not work for FO4 formulas. Actually the following FO4 formula is not expressible by any term of relation algebra (this result was published in [56] (according to [85, p.54. (iv)])):

$$\forall x.\forall y.\forall z.\exists w.\neg(x=w)\wedge\neg(y=w)\wedge\neg(z=w).$$

Axioms of Relation Algebra

Finally we remark on the axiomatizability of $\mathbf{RRA}_A$. The axioms of relation algebra for the calculus of relations was first introduced by Tarski in 1941 [84, p.76-77]. Subsequently Tarski gave an axiomatization by finitely many equational formulas in 1950s (Figure 2.5). An algebra $\mathcal{A}$ over σ^{RA} is called a *relation algebra* if $\mathcal{A}$ satisfies all

(r1) $a\cup(b\cup c)=(a\cup b)\cup c$
(r2) $b\cup a=b\cup a$
(r3) $(a^-\cup b^-)^-\cup(a^-\cup b)^-=a$
(r4) $a\cdot(b\cdot c)=(a\cdot b)\cdot c$
(r5) $a\cdot 1=a$
(r6) $(a\cup b)\cdot c=a\cdot c\cup b\cdot c$
(r7) $a^{\smile\smile}=a$
(r8) $(a\cup b)^\smile=a^\smile\cup b^\smile$
(r9) $(a\cdot b)^\smile=b^\smile\cdot a^\smile$
(r10) $a^\smile\cdot(a\cdot b)^-\cup b^-=b^-$
(r11) $a\cap b=(a^-\cup b^-)^-$
(r12) $\top=1\cup 1^-$
(r13) $0=\top^-$

FIGURE 2.5: Tarski's axioms of relation algebra (see e.g., [62, p.21.])

the axioms (r1)-(r13). (r1)-(r3) denotes that $(|\mathcal{A}|,\cup,\bullet^-)$ is a Boolean algebra. (r4)-(r5) denotes that $(|\mathcal{A}|,1,\cdot)$ is a monoid (note that the equation $1\cdot a=a$ can be derived by using (r5)(r7)(r9)). (r6) denotes the operation $\cdot$ distributes over the operation $\cup$

from the right. (r7)-(r9) are the axiom for the converse operation $\bullet^{\smile}$. (r10) is called Tarski/De Morgan law. (r11) (resp. (r12), (r13)) is the axiom for the symbols $\cap$ (resp. $\top$, 0). Let *RA* be the class of relation algebras and let $\mathbf{RA}_A$ be the equational theory defined by the class of relation algebras over an alphabet A. Every representable relation algebra satisfies all the axioms, and thus $\mathbf{RRA}_A \supseteq \mathbf{RA}_A$ (resp. $RRA \subseteq RA$) holds. In [84, p.87-88], Tarski asked whether his axioms are complete for $\mathbf{RRA}_A$ (and also asked whether every algebra satisfying his axioms is isomorphic to a proper relation algebra), but these have been solved negatively by Lyndon [61]. Concerning axiomatizability of $\mathbf{RRA}_A$, Tarski proved that *RRA* ($\mathbf{RRA}_A$) is (infinitely) axiomatizable by a set of equational formulas [83] and an explicit set of axioms is given by Lyndon [60]. Actually *RRA* ($\mathbf{RRA}_A$) is not finitely axiomatizable proved by Monk [66], where A is an infinite set. For more details about relation algebra, see the book "Relation Algebras" of Maddux [62].

2.3.2 Language Semantics

The language semantics of Kleene algebra is originated with Kleene's *regular events* (also called regular languages or regular sets) in 1950s [50]. Let $\mathcal{L}$, $\mathcal{L}_1$ and $\mathcal{L}_2$ be languages over A. The *concatenation* of $\mathcal{L}_1$ and $\mathcal{L}_2$, written $\mathcal{L}_1 \cdot \mathcal{L}_2$, is defined by $\mathcal{L}_1 \cdot \mathcal{L}_2 = \{s_1 \cdot s_2 \mid s_1 \in \mathcal{L}_1, s_2 \in \mathcal{L}_2\}$. The *n-th iterate* of $\mathcal{L}$, written $\mathcal{L}^n$, is inductively defined as follows: (1) $\mathcal{L}^0 = \{\varepsilon\}$; (2) $\mathcal{L}^{n+1} = \mathcal{L} \cdot \mathcal{L}^n$. The *Kleene star* of $\mathcal{L}$, written $\mathcal{L}^*$, is defined by $\mathcal{L}^* = \bigcup_{n \geq 0} \mathcal{L}^n$.

Definition 2.3.12 (Regular language)**.** The *class of regular languages* over an alphabet A, written REG_A, is the smallest class satisfying the follows:

(1) $\{a\} \in \mathsf{REG}_A$ for $a \in A$.

(2) $\emptyset \in \mathsf{REG}_A$.

(3) $\{\varepsilon\} \in \mathsf{REG}_A$.

(4) If $\mathcal{L}_1, \mathcal{L}_2 \in \mathsf{REG}_A$, then $\mathcal{L}_1 \cdot \mathcal{L}_2 \in \mathsf{REG}_A$.

(5) If $\mathcal{L}_1, \mathcal{L}_2 \in \mathsf{REG}_A$, then $\mathcal{L}_1 \cup \mathcal{L}_2 \in \mathsf{REG}_A$.

(6) If $\mathcal{L} \in \mathsf{REG}_A$, then $\mathcal{L}^* \in \mathsf{REG}_A$.

Definition 2.3.13 (Language semantics)**.** The *language* of a term t in $\mathcal{T}_A^{\mathsf{KA}}$, written $\mathcal{L}(t)$, is a language over A, inductively defined as follows:

(1) $\mathcal{L}(a) := \{a\}$ for $a \in A$;

(2) $\mathcal{L}(0) := \emptyset$;

(3) $\mathcal{L}(1) := \{\varepsilon\}$;

(4) $\mathcal{L}(t_1 \cdot t_2) := \mathcal{L}(t_1) \cdot \mathcal{L}(t_2)$;

(5) $\mathcal{L}(t_1 \cup t_2) := \mathcal{L}(t_1) \cup \mathcal{L}(t_2)$;

(6) $\mathcal{L}(t^*) := \mathcal{L}(t)^*$.

We call that a language $\mathcal{L}$ is *recognized* by a regular expression if there is a term $t \in \mathcal{T}_A^{\mathsf{KA}}$ such that $\mathcal{L} = \mathcal{L}(t)$. It is easy to see that $\mathcal{L} \in \mathsf{REG}_A$ if and only if $\mathcal{L}$ is recognized by a term in $\mathcal{T}_A^{\mathsf{KA}}$.

The equational theory on the language semantics, written $\mathbf{LKA}_A$, is a set of equational formulas, defined by $\mathbf{LKA}_A := \{t_1 = t_2 \in \Phi_A^{=,\mathsf{KA}} \mid \mathcal{L}(t_1) = \mathcal{L}(t_2)\}$. Actually $\mathbf{LKA}_A$ is equivalent to $\mathbf{RKA}_A$.

Theorem 2.3.14 (e.g., [52, Thm. 6], [68, Thm. VI.3], [72, Thm. 4])**.** $\mathbf{LKA}_A = \mathbf{RKA}_A$.

Finite Automata: NFAs and DFAs

In this subsubsection we define nondeterministic finite automata (NFAs) and deterministic finite automata (DFAs). It is well known that the class of languages recognizable by finite automata is equivalent to the class of regular languages (Kleene's theorem [50]).

Definition 2.3.15 (ε-NFA)**.** A nondeterministic finite automation with ε-transition (ε-NFA) $\mathcal{A}$ over an alphabet A is a tuple (Q, δ, q^I, F), where

(1) Q is a finite set denoting the set of *states*;

(2) $\delta\colon (A \cup \{\varepsilon\}) \to \wp(Q \times Q)$ denotes the *transition function* (every $\delta(a)$ is a binary relation on Q);

(3) $q^I \in Q$ denotes the *initial state*;

(4) $F \subseteq Q$ denotes the set of *acceptance states*.

$\hat{\delta}_{\mathcal{A}}(s)$ denotes the transition function of a string s combined with ε-transition. $\hat{\delta}_{\mathcal{A}}(s)$ is inductively defined satisfying the follows: (1) $\hat{\delta}_{\mathcal{A}}(\varepsilon) = \triangle(Q_{\mathcal{A}})$; (2) $\hat{\delta}_{\mathcal{A}}(a) = \delta_{\mathcal{A}}(\varepsilon)^* \cdot \delta_{\mathcal{A}}(a)$ for $a \in A$; (3) $\hat{\delta}_{\mathcal{A}}(as') = \hat{\delta}_{\mathcal{A}}(a) \cdot \hat{\delta}_{\mathcal{A}}(s')$. The *language* of an ε-NFA $\mathcal{A}$ over an alphabet A, written $\mathcal{L}(\mathcal{A})$, is defined by $\mathcal{L}(\mathcal{A}) = \{s \in A^* \mid (\hat{\delta}_{\mathcal{A}}(s) \cdot \delta_{\mathcal{A}}(\varepsilon)^*)(q^I) \cap F \neq \emptyset\}$. We call that a language $\mathcal{L}$ is *recognized* by an ε-NFA if there is an ε-NFA $\mathcal{A}$ such that $\mathcal{L} = \mathcal{L}(\mathcal{A})$. The *size* of an ε-NFA $\mathcal{A}$, written $\|\mathcal{A}\|$, is defined as $\#(Q_{\mathcal{A}})$.

Definition 2.3.16 (NFA)**.** An ε-NFA is called an *NFA* if $\delta_{\mathcal{A}}(\varepsilon) = \emptyset$.

Definition 2.3.17 (DFA)**.** An NFA $\mathcal{A}$ is called a *deterministic finite automaton (DFA)* if for any $a \in A$, $\delta_{\mathcal{A}}(a)$ is deterministic (i.e., $\#(\delta_{\mathcal{A}}(a)(q)) = 1$).

Remark. Every NFA over an alphabet A can be also regarded as a structure over the signature $(\{\delta_a\}_{a \in A}, q^I, F)$, where every δ_a is the binary relation symbol; q^I is the nullary function symbol; and F is the unary relation symbol. In fact we may regard every NFA as a structure in Chapter 4.

ε-NFA_A (resp. NFA_A, DFA_A) denotes the set of all ε-NFAs (resp. NFAs, DFAs) over the alphabet A. The following theorem is well known and there are many translations between these models (see e.g., [38]).

Theorem 2.3.18 (see e.g., [41, Sec. 3.2])**.** *Let A be an alphabet and let $\mathcal{L}$ be any language over A. Then the following are all equivalent:*

(1) $\mathcal{L}$ is recognized by a regular expression in $\mathcal{T}_A^{\mathsf{KA}}$;

(2) $\mathcal{L}$ is recognized by an ε-NFA in ε-NFA_A;

(3) $\mathcal{L}$ is recognized by an NFA in NFA_A;

(4) $\mathcal{L}$ is recognized by a DFA in DFA_A.

Regular expressions, NFAs, and DFAs all recognize the class of regular languages. However these models are different with respect to the succinctness. Figure 2.6 summarizes translation bounds for these models. Dotted arrows are used for trivial translations. Every black arrow denotes that there is a polynomial-time translation, every red arrow denotes that there is an exponential blowup, every black colored

term denotes the upperbound of the size, and every red colored term denotes the lower bound of the size.

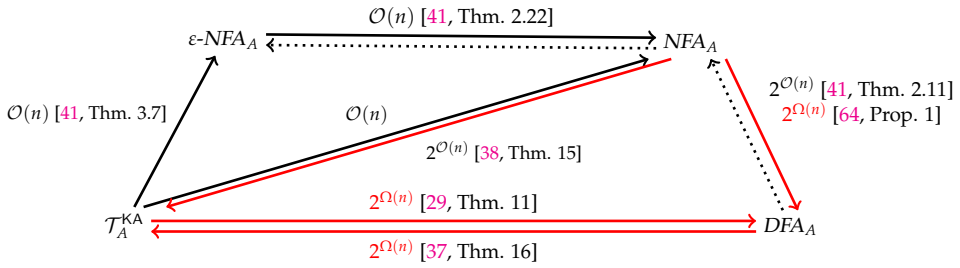


FIGURE 2.6: An overview of translation bounds for models for regular languages, where $\#A \geq 2$

Derivatives (on Strings)

The (partial) derivatives on strings developed by Brzozowski [19] and Antimirov [3] are basis of the (partial) derivatives on graphs in Chapter 3. Derivatives give a translation from regular expressions to finite automata. In this thesis, we employ Antimirov' notation [3] because the algorithm obtained from the derivatives is easy to analyze computational complexity (see Proposition 2.3.25).

We use $T, T', \ldots$ to denote a set of terms. The concatenation of T_1 and T_2, is defined by $T_1 \cdot T_2 = \{t_1 \cdot t_2 \mid t_1 \in T_1, t_2 \in T_2\}$. $\mathcal{L}(T)$ denotes the language $\bigcup_{t \in T} \mathcal{L}(t)$.

Derivatives on strings are deeply related to the following operations.

Definition 2.3.19 (Left quotients on strings)**.** The *left quotient* of a language $\mathcal{L}$ over an alphabet A with respect to a string s, written $s^{-1}\mathcal{L}$, is a language, defined as $\{s' \in A^* \mid ss' \in \mathcal{L}\}$.

Derivatives on strings consist of *empty string property* E and *partial derivatives on characters* D_a.

Definition 2.3.20 (Empty string property (Reg1 in [3]))**.** The *empty string property* of a term t in $\mathcal{T}_A^{\mathsf{KA}}$, written $E(t)$, is a truth value, inductively defined as follows:

(1) $E(a) := false$;

(2) $E(0) := false$;

(3) $E(1) := true$;

(4) $E(t_1 \cdot t_2) := E(t_1) \wedge E(t_2)$;

(5) $E(t_1 \cup t_2) := E(t_1) \vee E(t_2)$;

(6) $E(t^*) := \mathit{true}$.

Definition 2.3.21 (Derivatives on characters [3])**.** The *derivative* with respect to a character $a \in A$, written D_a, is a function $\mathcal{T}_A^{\mathsf{KA}} \to \wp(\mathcal{T}_A^{\mathsf{KA}})$ (i.e., a binary relation on $\mathcal{T}_A^{\mathsf{KA}}$), inductively defined satisfying the follows:

(1) $D_a(a') := \begin{cases} \{1\} & (a' = a) \\ \emptyset & (\mathit{otherwise}) \end{cases}$;

(2) $D_a(0) := \emptyset$;

(3) $D_a(1) := \emptyset$;

(4) $D_a(t_1 \cup t_2) := D_a(t_1) \cup D_a(t_2)$;

(5) $D_a(t_1^*) := D_a(t_1) \cdot \{t_1^*\}$;

(6) $D_a(t_1 \cdot t_2) := \begin{cases} D_a(t_1) \cdot \{t_2\} \cup D_a(t_2) & (E(t_1)) \\ D_a(t_1) \cdot \{t_2\} & (\mathit{otherwise}) \end{cases}$.

Definition 2.3.22 (Derivatives on strings [3])**.** The *derivative* with respect to a string $s \in A^*$, written D_s, is a function $\mathcal{T}_A^{\mathsf{KA}} \to \wp(\mathcal{T}_A^{\mathsf{KA}})$ (i.e., a binary relation on $\mathcal{T}_A^{\mathsf{KA}}$), defined inductively as follows: $D_s = \triangle(\mathcal{T}_A^{\mathsf{KA}})$ if $s = \varepsilon$, and $D_s = D_{s'} \cdot D_a$ if $s = s'a$.

$E(T)$ denotes the truth value $\bigvee_{t \in T} E(t)$, and $D_s(T)$ denotes the set of terms $\bigcup_{t \in T} D_s(t)$.

The next proposition shows left quotients can be characterized by the derivatives.

Proposition 2.3.23 ([3, Prop. 2.10])**.**

(1) $E(t) \iff \varepsilon \in L(t)$.

(2) $\mathcal{L}(D_s(t)) = s^{-1}L(t)$.

(3) $E(D_s(t)) \iff s \in \mathcal{L}(t)$.

The derivatives give some algorithms for language problems. For example, the *membership problem* (i.e., given a term t and a string s, $s \in \mathcal{L}(t)$?) is determined by checking whether $E(D_s(t))$ holds or not. Moreover, problems with searching a string, for example, the *(language) inclusion problem* (i.e., given two terms, t_1 and t_2,

$\mathcal{L}(t_1) \subseteq \mathcal{L}(t_2)$?) and the *universality problem* (i.e., given a term t_1, $\mathcal{L}(t_1) = A^*$?) can be also determined by using the derivatives. Proposition 2.3.25 enables us to restrict the search space to be finite.

Definition 2.3.24. The *closure* of a term t in $\mathcal{T}_A^{\mathsf{KA}}$, written $\mathrm{cl}(t)$, is a set of terms, defined inductively as follows:

(1) $\mathrm{cl}(0) := \{0\}$;

(2) $\mathrm{cl}(1) := \{1\}$;

(3) $\mathrm{cl}(a) := \{1, a\}$;

(4) $\mathrm{cl}(t_1^*) := \mathrm{cl}(t_1) \cdot \{t_1^*\} \cup \{t_1^*\}$;

(5) $\mathrm{cl}(t_1 \cup t_2) := \mathrm{cl}(t_1) \cup \mathrm{cl}(t_2) \cup \{t_1 \cup t_2\}$;

(6) $\mathrm{cl}(t_1 \cdot t_2) := \mathrm{cl}(t_1) \cdot \{t_2\} \cup \mathrm{cl}(t_2) \cup \{t_1 \cdot t_2\}$.

$\mathrm{cl}(T)$ denotes the set of terms $\bigcup_{t \in T} \mathrm{cl}(t)$. cl is a *closure operator*, i.e., cl satisfies (a) $T \subseteq \mathrm{cl}(T)$; (b) $T_1 \subseteq T_2 \Longrightarrow \mathrm{cl}(T_1) \subseteq \mathrm{cl}(T_2)$; and (c) $\mathrm{cl}(\mathrm{cl}(T)) = \mathrm{cl}(T)$.

Proposition 2.3.25 ([3, Thm. 3.4])**.**

(1) $D_s(t) \subseteq \mathrm{cl}(t)$.

(2) $\#(\mathrm{cl}(t)) \leq 1 + \|t\|$.

By Proposition 2.3.25, it is enough to consider terms in $\mathrm{cl}(t)$ for calculating $D_s(t)$ from a given term t. It is because the following hold for any string $s = a_1 \ldots a_n$: (1) $D_s(t) = (D_s \restriction \mathrm{cl}(t))(t)$ (by Proposition 2.3.25(1)); (2) $D_s \restriction \mathrm{cl}(t) = D_{a_1} \restriction \mathrm{cl}(t) \cdot \ldots \cdot D_{a_n} \restriction \mathrm{cl}(t)$ (by Proposition 2.3.25(1) and that cl is a closure operator), where $f \restriction X$ denotes the *restriction* of a (partial) function f. Under the restriction, the number of patterns of binary relations on $\mathrm{cl}(t)$ is at most $2^{\#(\mathrm{cl}(t))^2}$. The finiteness justifies Algorithm 1.

Theorem 2.3.26 (e.g., [3])**.** *The language inclusion problem for* $\mathcal{T}_A^{\mathsf{KA}}$ *is in* PSPACE.

Proof. By Algorithm 1 and Proposition 2.3.25(2), the problem is in coNPSPACE, and hence in PSPACE. (Note that coNPSPACE = PSPACE by Savitch's theorem [77].) □

Algorithm 1 The language inclusion problem for $\mathcal{T}_A^{\mathsf{KA}}$

Ensure: $\mathcal{L}(t_1) \subseteq \mathcal{L}(t_2)$?
  $CL \Leftarrow \mathrm{cl}(t_1) \cup \mathrm{cl}(t_2)$
  $(D, d) \Leftarrow (\triangle(CL), 0)$
  while $d < 2^{\#(CL)^2}$ **do**
    if $E(D(t_1)) \wedge \neg E(D(t_2))$ **then**
      return $false$
    end if
    pickup $a \in A$ nondeterministically
    $(D, d) \Leftarrow (D \cdot D_a, d + 1)$
  end while
  return $true$

(In fact the problem is PSPACE-complete, see e.g., [43, Prop. 2.4].) By using derivatives on strings, any language of each term applied any left quotient is representable by finitely many 'sub'terms, whereas the language may be infinite.

Corollary 2.3.27. $\mathbf{LKA}_A$ *(resp.* $\mathbf{RKA}_A$*) is decidable in* PSPACE.

Axioms of Kleene Algebra

Kleene gave some properties for the equational theory $\mathbf{LKA}_A$ ($\mathbf{RKA}_A$) in [50, Sec. 7.2] and posed axiomatization as an open problem. Redko proved that $\mathbf{LKA}_A$ is not finitely axiomatizable by equational formulas [73][25, Thm. 9]. A finite axiomatization is given by Salomaa [75], but the axiomatization uses a non equational formula [75, "R2": $(a = a \cdot b \cup c \wedge \neg(b \geq 1)) \rightarrow a = c \cdot b^*$]. In 1991, Kozen gave a finite quasi-equational axiomatization [51], which solves the conjecture posed by Conway [25, p.103]. *Quasi-equational axiomatization* is axiomatization using formulas of the form $E_1 \wedge E_2 \wedge \cdots \wedge E_n \rightarrow E$ (called *universal horn formula*), where each $E_1, \ldots E_n, E$ is an equational formula. Figure 2.7 shows Kozen's axiomatization for $\mathbf{LKA}_A$ [51]. Let KA be the class of structures over σ^{KA} satisfying the axioms in Figure 2.7. As mentioned above, this axiomatization is complete.

Theorem 2.3.28 ([51])**.** $\mathbf{KA}_A = \mathbf{LKA}_A$*, where* $\mathbf{KA}_A$ *is the equational theory defined by* KA*, i.e.,* $\mathbf{KA}_A := \{t_1 = t_2 \in \Phi_A^{=,\mathsf{KA}} \mid \mathcal{A} \models t_1 = t_2$ *for any structure* $\mathcal{A} \in KA\}$.

We remark that there is also an axiomatization for universal horn formulas, and thus there is a (finite) proof system for Kleene algebra. Figure 2.8 shows the axioms and the deduction rules for universal horn formulas based on [79, Thm. 2]. (Figure 2.8

(k1) $a \cup (b \cup c) = (a \cup b) \cup c$
(k2) $b \cup a = b \cup a$
(k3) $a \cup 0 = a$
(k4) $a \cup a = a$
(k5) $a \cdot (b \cdot c) = (a \cdot b) \cdot c$
(k6) $1 \cdot a = a$
(k7) $a \cdot 1 = a$
(k8) $a \cdot (b \cup c) = a \cdot b \cup a \cdot c$
(k9) $(a \cup b) \cdot c = a \cdot c \cup b \cdot c$
(k10) $0 \cdot a = 0$
(k11) $a \cdot 0 = 0$
(k12) $1 \cup a \cdot a^* = a^*$
(k13) $1 \cup a^* \cdot a = a^*$
(k14) $a \cdot x \leq x \to a^* \cdot x \leq x$
(k15) $x \cdot a \leq x \to x \cdot a^* \leq x$

where $a \leq b$ denotes the equational formula $a \cup b = b$.

FIGURE 2.7: The quasi-equational axioms of Kleene algebra [51]

is some modified from [79, Thm. 2] like the sequent calculus for intuitionistic logic, but not essential. $E_1 \wedge E_2 \wedge \cdots \wedge E_n \to E$ denotes the formula E if $n = 0$.)

(u1) $a = a$
(u2) $a = a' \to a' = a$
(u3) $a_1 = a_2 \wedge a_2 = a_3 \to a_1 = a_3$
(u4) $a_1 = a'_1 \wedge \cdots \wedge a_{\mathrm{ar}(f)} = a'_{\mathrm{ar}(f)} \to f(a_1, \ldots, a_{\mathrm{ar}(f)}) = f(a'_1, \ldots, a'_{\mathrm{ar}(f)})$
(u5) $\dfrac{E_1 \wedge \cdots \wedge E_n \to E}{E_1 \wedge \cdots \wedge E_n \wedge E' \to E}$ (w)
(u6) $\dfrac{E_1 \wedge \cdots \wedge E_n \wedge E' \wedge E' \to E}{E_1 \wedge \cdots \wedge E_n \wedge E' \to E}$ (c)
(u7) $\dfrac{E_1 \wedge \ldots E_i \ldots E_j \cdots \wedge E_n \to E}{E_1 \wedge \ldots E_j \ldots E_i \cdots \wedge E_n \to E}$ (e)
(u8) $\dfrac{E_1 \wedge \cdots \wedge E_n \to E \qquad E \wedge E'_1 \wedge \cdots \wedge E'_m \to E'}{E_1 \wedge \cdots \wedge E_n \wedge E'_1 \wedge \cdots \wedge E'_m \to E'}$ (cut)
(u9) $\dfrac{\varphi}{\varphi[t/a]}$ (assignment)

where $n, m \geq 0$ and $[t/a]$ denotes the assignment (i.e., $\varphi[t/a]$ denotes φ in which each occurrence of a has been replaced by t).

FIGURE 2.8: An axiomatization for universal horn formulas

Moreover there is a cut-free proof system for Kleene algebra [27]. The proof system is non-wellfounded, but enables to construct a proof search.

Chapter 3

Derivatives for Kleene Allegories

(This chapter is based on the author's papers [68, 70].)

In this chapter we show that the equational theory of *Kleene allegories* is decidable and EXPSPACE-complete. This problem was open in the paper [18] of Brunet and Pous. The proof proceeds by designing *derivatives on graphs*, which are generalizations of derivatives on strings for regular expressions, called Antimirov' (partial) derivatives [3]. The derivatives on graphs give a finite automata construction algorithm as with the derivatives on strings.

The signature of Kleene allegories, written σ^{KAI}, consists of the constant symbols 0 and 1, the unary function symbols $\bullet^*$ and $\bullet^\smile$, and the binary function symbols $\cdot$, $\cup$, and $\cap$. $\mathcal{T}_A^{\mathsf{KAI}}$ denotes the set of terms of Kleene allegories. Kleene allegories subsume (relational) Kleene algebras and (representable distributive) allegories (or called the positive calculus of relations [72]). Their equational theories are decidable. (see Section 2.3.2 for Kleene algebras and see [32] for (representable) allegories.) Especially, *identity-free Kleene lattice* is decidable and EXPSPACE [17]. The signature of identity-free Kleene lattice consists of the constant symbol 0, the unary function symbols $\bullet^+$, and the binary function symbols relations $\cdot$, $\cup$, and $\cap$. (The symbol $\bullet^+$ denotes the transitive closure operator.) However the decidability and computational complexity of Kleene allegories was open. Our main contribution is to resolve the open question positively. The idea is based on *derivatives* on strings for regular expressions, that are tools to obtain the decidability of decision problems for regular expressions (see Section 2.3.2).

In this work, we extend the derivatives from strings to graphs, and show the decidability of Kleene allegories by the following steps: (1) we extend some existing definitions with "labels"; this extension is effective for defining partial derivatives on graphs (Section 3.2); (2) we design procedures for constructing finite graphs (called "Sequential Graph Construction Procedures", or SGCPs for short) by using labels (Section 3.3); (3) we give derivatives on graphs constructed by SGCPs, and show that the equational theory is also decidable in EXPSPACE (Section 3.5).

3.1 Kleene Allegories

In this section we introduce relational semantics and graph language semantics of Kleene allegories.

3.1.1 (Relational) Semantics

The (in)equational theory of *Kleene allegories* is defined by using relational semantics.

Definition 2.3.1 (restated). A *relational model* M (over an alphabet A) is a tuple $(V, \{R_a\}_{a\in A})$, where V is a set; and every $R_a \subseteq V \times V$ is a binary relation on V.

Definition 3.1.1 (Relational Semantics)**.** The *relation* of a term t in $\mathcal{T}_A^{\mathsf{KAI}}$ with respect to a relational model $M = (V, \{R_a\}_{a\in A})$, written $[\![t]\!]_M$ is inductively defined as follows:

(1) $[\![a]\!]_M = R_a$;

(2) $[\![0]\!]_M = \emptyset$;

(3) $[\![1]\!]_M = \triangle(V)$;

(4) $[\![t_1 \cdot t_2]\!]_M = [\![t_1]\!]_M \cdot [\![t_2]\!]_M$;

(5) $[\![t_1 \cup t_2]\!]_M = [\![t_1]\!]_M \cup [\![t_2]\!]_M$;

(6) $[\![t_1 \cap t_2]\!]_M = [\![t_1]\!]_M \cap [\![t_2]\!]_M$;

(7) $[\![t^*]\!]_M = [\![t]\!]_M^*$;

(8) $[\![t^\smile]\!]_M = [\![t]\!]_M^\smile$.

$M \models t_1 = t_2$ (resp. $M \models t_1 \leq t_2$) denotes $[\![t_1]\!]_M = [\![t_2]\!]_M$ (resp. $[\![t_1]\!]_M \subseteq [\![t_2]\!]_M$). REL $\models t_1 \leq t_2$ denotes that $M \models t_1 \leq t_2$ holds for any relation model M.

Definition 3.1.2 (Homomorphism)**.** Let $M_i = (V^i, \{R^i_a\}_{a \in A})$ be a relation model for $i = 1, 2$. A function $h : V^2 \to V^1$ is called a *homomorphism* from M_2 to M_1 if, for any nodes, $v, v' \in V^2$ and any character a,

$$(v, v') \in R^2_a \Longrightarrow (h(v), h(v')) \in R^1_a.$$

$M_1 \triangleleft M_2$ denotes there is a homomorphism from M_2 to M_1.

Let us recall that the two equational theories defined by language semantics and by relational semantics are equivalent for terms of Kleene algebra (see Theorem 2.3.14). However, when we add intersection ($\cap$) or converse ($\bullet^\smile$) to terms of Kleene algebra, the equivalence does not hold, where the (string) language semantics is defined as Definition 2.3.13 with the following rules: $L(t_1 \cap t_2) = L(t_1) \cap L(t_2)$ and $L(t^\smile) = \{a_n \dots a_1 \mid a_1 \dots a_n \in L(t)\}$. Let consider the next examples: (a) $a \cap aa = a \cap aaa$; (b) $a = a^\smile$; and (c) $a \leq aa^\smile a$. Each of (a) and (b) holds in language semantics, however does not in relational semantics. In (a), the terms of both sides interpret the empty language in language semantics, however it is easy to construct a relation model such that these terms are not equivalent [2, (1.8)]. In (b), the terms of both sides interpret the singleton language $\{a\}$, however the equation does not hold unless the relation of the character a on a relational model is symmetric [18]. In contrast, (c) holds in relational semantics, however does not in language semantics [11, (10)].

Although the converse $\bullet^\smile$ in relational semantics cannot be completely erased (like the converse $\bullet^\smile$ in language semantics), any terms can be modified so that $\bullet^\smile$ is only applied to characters, by using the following rewriting rule [18]:

$$1^\smile \rightsquigarrow 1;\ 0^\smile \rightsquigarrow 0;\ (t_1 \cdot t_2)^\smile \rightsquigarrow t_2^\smile \cdot t_1^\smile;\ (t^*)^\smile \rightsquigarrow (t^\smile)^*;$$
$$(t_1 \cup t_2)^\smile \rightsquigarrow t_1^\smile \cup t_2^\smile;\ (t_1 \cap t_2)^\smile \rightsquigarrow t_1^\smile \cap t_2^\smile;\ t^{\smile\smile} \rightsquigarrow t.$$

For that reason, we assume that the set of terms is defined by the following grammar:

(Terms) $t ::= 0 \mid 1 \mid a \mid a^\smile \mid t_1 \cdot t_2 \mid t_1 \cup t_2 \mid t_1 \cap t_2 \mid t_1^*$

3.1.2 Graph Languages

Each term of Kleene allegories expresses a set of graphs with a single source and a single target. The inclusion problem for Kleene allegories (REL $\models t_1 \leq t_2$?) can be solved by comparing their graph languages (Theorem 3.2.9).

We define some mathematical notations needed later. $X_1 \uplus X_2$ denotes the *disjoint union* of two sets, X_1 and X_2, defined by $X_1 \uplus X_2 = \{(x_1, 1) \mid x_1 \in X_1\} \cup \{(x_2, 2) \mid x_2 \in X_2\}$. When i is clear from the context, (x_i, i) may be abbreviated as x_i. The *equivalence class* of an element x on X with a equivalence relation Θ, written $[x]_\Theta$, is defined by $[x]_\Theta = \{x' \in X \mid (x, x') \in \Theta\}$. When Θ is clear from the context, the subscript Θ may be abbreviated. Then the *quotient set* of X by Θ, written X/Θ, is defined as $\{[x] \mid x \in X\}$. $[X']$ denotes the set $\{[x] \mid x \in X'\}$. $R^=$ denotes the *equivalence closure* (reflexive symmetric transitive closure) of R, i.e., $R^= = (R \cup R^\smile)^*$.

Let $\check{a}$ be a character denoting the converse of a character a and $\mathtt{1}$ be the special constant character. $\check{A}$ denotes the set $\{\check{a} \mid a \in A\}$, $\check{A}_1$ denotes $\check{A} \cup \{\mathtt{1}\}$, $\dot{A}$ denotes $A \cup \check{A}$, and $\dot{A}_1$ denotes $\dot{A} \cup \{\mathtt{1}\}$.

Definition 3.1.3. A *graph* is a tuple $G = (V, \{R_{\dot{a}}\}_{\dot{a} \in \dot{A}_1}, v_s, v_t)$, where V is a nonempty set of nodes, each $R_{\dot{a}}$ is a relation on V, $v_s \in V$ is a node called the *source node*, and $v_t \in V$ is a node called the *target node*.

A graph G is said to be *normal formed* if $R_{\check{A}_1} = \emptyset$. G may be denoted as (M, v_s, v_t) by using a relation model M. $M_1 \uplus M_2$ denotes the *disjoint union of relation models*, defined as $(V^{M_1} \uplus V^{M_2}, \bigcup_{i \in \{1,2\}} \{((v_i, i), (v_i', i)) \mid (v_i, v_i') \in R^{M_i}\})$ and M/Θ denotes the *quotient relation model*, defined as $(V^M/\Theta, \{([x], [x']) \mid (x, x') \in R^M\})$.

The *series-composition* of G_1 and G_2, written $G_1 \cdot G_2$, is a graph, defined as $((M^{G_1} \uplus M^{G_2})/\Theta, [v_s^{G_1}], [v_t^{G_2}])$, where $\Theta = \{(v_t^{G_1}, v_s^{G_2})\}^=$.

The *parallel-composition* of G_1 and G_2, written $G_1 \parallel G_2$, is a graph, defined as $((M^{G_1} \uplus M^{G_2})/\Theta, [v_s^{G_1}], [v_t^{G_1}])$, where $\Theta = \{(v_s^{G_1}, v_s^{G_2}), (v_t^{G_1}, v_t^{G_2})\}^=$.

A *graph language* $\mathcal{G}$ is a set of graphs. $\mathcal{G}_1 \bullet \mathcal{G}_2$ denotes the set of graphs $\{G_1 \bullet G_2 \mid G_1 \in \mathcal{G}_1, G_2 \in \mathcal{G}_2\}$ for $\bullet \in \{\cdot, \|\}$.

Definition 3.1.4 (Graph languages)**.** The *graph language* of a term $\mathcal{G}(t)$ is defined inductively as follows:

$$\mathcal{G}(1) := \left\{ \rightarrow\circ\xrightarrow{1}\circ\rightarrow \right\}; \mathcal{G}(\dot{a}) := \left\{ \rightarrow\circ\xrightarrow{\dot{a}}\circ\xrightarrow{1}\circ\rightarrow \right\};$$

$$\mathcal{G}(0) := \emptyset;\ \mathcal{G}(t_1 \cdot t_2) := \mathcal{G}(t_1) \cdot \mathcal{G}(t_2);$$

$$\mathcal{G}(t_1^*) := \mathcal{G}(1) \cup \{G_1 \cdot \ldots \cdot G_n \mid n \geq 1, \forall i. G_i \in \mathcal{G}(t_1)\};$$

$$\mathcal{G}(t_1 \cup t_2) := \mathcal{G}(t_1) \cup \mathcal{G}(t_2);\ \mathcal{G}(t_1 \cap t_2) := \mathcal{G}(t_1) \parallel \mathcal{G}(t_2).$$

Any graph in the graph language of any term is a (directed) series-parallel graph [30].

Remark. $\mathcal{G}(\dot{a})$ in the above definition may seem strange since the graph in the definition have an extra edge with the constant label 1. However, by the modification, some discussions are simplified because any labelled graph in a labelled graph language are "simple" (defined in Definition 3.4.1).

Definition 3.1.5. The *normal form* of a graph G, written $\langle\langle G \rangle\rangle$, is defined by $\langle\langle G \rangle\rangle := ((V^G, \{R_{\dot{a}}\}_{\dot{a} \in \dot{A}_1})/(R_1^G)^=, [v_s^G], [v_t^G])$, where $R_a = R_a^G \cup R_{\breve{a}}^{G^\smile}$ if $a \in A$, and $R_{\dot{a}} = \emptyset$ if $\dot{a} \in \breve{A} \cup \{1\}$.

$\langle\langle \mathcal{G} \rangle\rangle$ denotes $\{\langle\langle G \rangle\rangle \mid G \in \mathcal{G}\}$. Note that $\langle\langle \mathcal{G}(t) \rangle\rangle$ is coincides with the graph language of t defined in [18].

Definition 3.1.6 (Homomorphism [18])**.** Let G_i be a *normal formed* graph for $i = 1, 2$. A function $h \colon V^{G_2} \to V^{G_1}$ is called a *homomorphism* from G_2 to G_1 if the following are satisfied: (a) h is a homomorphism from M^{G_2} to M^{G_1}; (b) $h(v_s^{G_2}) = v_s^{G_1}$; (c) $h(v_t^{G_2}) = v_t^{G_1}$.

$G_1 \triangleleft G_2$ denotes that there exists a homomorphism from $\langle\langle G_2 \rangle\rangle$ to $\langle\langle G_1 \rangle\rangle$, where G_i is a graph for $i = 1, 2$. $\mathcal{G}_1 \triangleleft \mathcal{G}_2$ denotes that, for any graph $G_1 \in \mathcal{G}_1$, there exists a graph $G_2 \in \mathcal{G}_2$ such that $G_1 \triangleleft G_2$.

Theorem 3.1.7 ([18, Thm. 6] (see also [1, Thm. 1]))**.**

$$\mathrm{REL} \models t_1 \leq t_2 \iff \mathcal{G}(t_1) \triangleleft \mathcal{G}(t_2).$$

In the next section, we extend the above theorem with labels.

3.2 Extension with Labels

We extend relation models, terms, and graph languages with labels. In a nutshell, the extension is for expressing terms and graphs having multiple source nodes. This extension is effective to define Sequential Graph Construction Procedures and derivatives on graphs in later sections.

Let L be a set of *labels*, and $l \in L$ denotes a label. The set of *labelled terms* $\tilde{\mathcal{T}}$ is defined by the following grammar:

(Terms) $t ::= 0 \mid 1 \mid a \mid a^\smile \mid t_1 \cdot t_2 \mid t_1 \cup t_2 \mid t_1 \cap t_2 \mid t_1^*$

(Labelled Terms) $\tilde{t} ::= @l.t \mid \tilde{t}_1 \cdot t_2 \mid \tilde{t}_1 \cap \tilde{t}_2$

Intuitively $@l.t$ denotes t in which the start point of t is forcibly regarded as the node with the label l. This is used for fixing the source nodes of terms and expressing terms which have multiple source nodes. This notation is derived from the jump operator in hybrid logics [9, p.49]. $A(\tilde{t})$ denotes the set of all characters occurring in $\tilde{t}$, and $L(\tilde{t})$ denotes the set of all labels occurring in $\tilde{t}$. We use $\tilde{T}$ to denote a set of labelled terms. $\tilde{\mathcal{T}}_L$ denotes the set $\{\tilde{t} \in \tilde{\mathcal{T}} \mid L(\tilde{t}) \subseteq L\}$, $\tilde{T}_1 \cdot T_2$ denotes the set $\{\tilde{t}_1 \cdot t_2 \mid \tilde{t}_1 \in \tilde{T}_1, t_2 \in T_2\}$, and $\tilde{T}_1 \Cap \tilde{T}_2$ denotes the set $\{\tilde{t}_1 \cap \tilde{t}_2 \mid \tilde{t}_1 \in \tilde{T}_1, \tilde{t}_2 \in \tilde{T}_2\}$.

Definition 3.2.1 (Labelled relation models)**.** A *labelled relation model* $\tilde{M}$ is a tuple (M, m), where $M = (V, \{R_a\}_{a \in A})$ is a relation model and m is an injective and partial function from L to V. $L(\tilde{M})$ denotes the domain of m. The relation of a labelled term $\tilde{t}$ with respect to $\tilde{M}$, written $R_{\tilde{t}}^{\tilde{M}}$, is defined inductively as follows:

$$(v, v') \in R_{@l.t}^{\tilde{M}} :\Leftrightarrow m(l) \text{ is defined and } (m(l), v') \in R_t^M;$$
$$R_{\tilde{t}_1 \cdot t_2}^{\tilde{M}} := R_{\tilde{t}_1}^{\tilde{M}} \cdot R_{t_2}^M;\ R_{\tilde{t}_1 \cap \tilde{t}_2}^{\tilde{M}} := R_{\tilde{t}_1}^{\tilde{M}} \cap R_{\tilde{t}_2}^{\tilde{M}}.$$

$\tilde{M} \models \tilde{t}_1 \leq \tilde{t}_2$ denotes $R_{\tilde{t}_1}^{\tilde{M}} \subseteq R_{\tilde{t}_2}^{\tilde{M}}$, and $\mathrm{REL}^\smile \models \tilde{t}_1 \leq \tilde{t}_2$ denotes that, for any labelled relation model $\tilde{M}$, $\tilde{M} \models \tilde{t}_1 \leq \tilde{t}_2$.

It is easy to see the next proposition holds.

Proposition 3.2.2. *For any label l_s,*

$$\mathrm{REL}^{\smile} \models @l_s.t_1 \leq @l_s.t_2 \iff \mathrm{REL} \models t_1 \leq t_2.$$

Sketch. It is shown by that the following hold for any M:
$(M,m) \models @l_s.t_1 \leq @l_s.t_2 \Leftrightarrow (M, m{\restriction}\{l_s\}) \models @l_s.t_1 \leq @l_s.t_2$;
$M \models t_1 \leq t_2 \Leftrightarrow \forall v_s.(M, \{l_s \mapsto v_s\}) \models @l_s.t_1 \leq @l_s.t_2$. □

Definition 3.2.3 (Labelled graphs)**.** A *labelled graph* $\tilde{G}$ is a tuple $(V, \{R_{\dot{a}}\}_{\dot{a}\in\dot{A}_1}, m, v_t)$, where V is a nonempty set of nodes, each $R_{\dot{a}}$ is a relation on V; m is a partial function from L to $\wp(V)$ such that m is injective viewed as a binary relation over L and V (i.e., if $(l,v) \in m$ and $(l',v) \in m$, then $l = l'$); and $v_t \in V$ is a node called the *target node*. $\mathrm{dom}(m)$ and $\mathrm{cod}(m)$ denote the *domain* and the *codomain* of m, i.e., $\mathrm{dom}(m) = \{l \mid \exists v.(l,v) \in m\}$ and $\mathrm{cod}(m) = \{v \mid \exists l.(l,v) \in m\}$, respectively. $L(\tilde{G})$ denotes $\mathrm{dom}(m^{\tilde{G}})$. $\tilde{G}$ may be denoted as (M, m, v_t) by using a relation model M, or denoted as $(\tilde{M}, v_t)$ by using a labelled relation model $\tilde{M}$ when $m^{\tilde{G}}$ is functional (i.e., if $(l,v) \in m^{\tilde{G}}$ and $(l,v') \in m^{\tilde{G}}$, then $v = v'$).

The *series-composition* of $\tilde{G}_1$ and G_2, written $\tilde{G}_1 \cdot G_2$, is defined by $\tilde{G}_1 \cdot G_2 := ((M^{\tilde{G}_1} \uplus M^{G_2})/\{(v_t^{\tilde{G}_1}, v_s^{G_2})\}^{=}, m, [v_t^{G_2}])$, where m is defined by $m(l) = [m^{\tilde{G}_1}(l)]$.

The *parallel-composition* of $\tilde{G}_1$ and $\tilde{G}_2$, written $\tilde{G}_1 \parallel \tilde{G}_2$, is defined by $\tilde{G}_1 \parallel \tilde{G}_2 := ((M^{\tilde{G}_1} \uplus M^{\tilde{G}_2})/\{(v_t^{\tilde{G}_1}, v_t^{\tilde{G}_2})\}^{=}, m, [v_t^{\tilde{G}_1}])$, where m is defined by $m(l) = [m^{\tilde{G}_1}(l) \cup m^{\tilde{G}_2}(l)]$.

A *labelled graph language* $\tilde{\mathcal{G}}$ is a set of labelled graphs. $\tilde{\mathcal{G}}_1 \cdot \mathcal{G}_2$ denotes $\{\tilde{G}_1 \cdot G_2 \mid \tilde{G}_1 \in \tilde{\mathcal{G}}_1, G_2 \in \mathcal{G}_2\}$ and $\tilde{\mathcal{G}}_1 \parallel \tilde{\mathcal{G}}_2$ denotes $\{\tilde{G}_1 \parallel \tilde{G}_2 \mid \tilde{G}_1 \in \tilde{\mathcal{G}}_1, \tilde{G}_2 \in \tilde{\mathcal{G}}_2\}$. $\tilde{\mathfrak{G}}$ denotes the set of all labelled graphs and $\tilde{\mathfrak{G}}_{L'}$ denotes the set $\{\tilde{G} \in \tilde{\mathfrak{G}} \mid L(\tilde{G}) \subseteq L'\}$.

Definition 3.2.4. The *labelled graph language of a labelled term* $\tilde{t}$, written $\tilde{\mathcal{G}}(\tilde{t})$, is defined inductively as follows:

$$\tilde{\mathcal{G}}(@l.t) := \{(g, \{(l, v_s)\}, v_t) \mid (g, v_s, v_t) \in \mathcal{G}(t)\};$$
$$\tilde{\mathcal{G}}(\tilde{t}_1 \cdot t_2) := \tilde{\mathcal{G}}(\tilde{t}_1) \cdot \mathcal{G}(t_2);\ \tilde{\mathcal{G}}(\tilde{t}_1 \cap \tilde{t}_2) := \tilde{\mathcal{G}}(\tilde{t}_1) \parallel \tilde{\mathcal{G}}(\tilde{t}_2).$$

$\tilde{G}$ is said to be *normal formed* if $R^{\tilde{G}}_{\check{A}_1} = \emptyset$ and $m^{\tilde{G}}$ is functional.

Definition 3.2.5. The *normal form* of a labelled graph $\tilde{G}$, written $\langle\langle \tilde{G} \rangle\rangle$, is defined as $((V^G, \{R_{\dot{a}}\}_{\dot{a} \in \dot{A}_1})/\Theta, m, [v_t^{\tilde{G}}])$, where (a) $\Theta = (\bigcup_{l \in L}(m^{\tilde{G}}(l) \times m^{\tilde{G}}(l)) \cup R_1^{\tilde{G}})^=$; (b) $R_a = R_a^{\tilde{G}} \cup R_{\breve{a}}^{\tilde{G}^\smile}$ if $a \in A$, and $R_{\dot{a}} = \emptyset$ if $\dot{a} \in \breve{A}_1$; and (c) $m(l) = [m^{\tilde{G}}(l)]$.

$\langle\langle \tilde{G} \rangle\rangle$ may be undefined since m may not be injective in the above definition. $\langle\langle \tilde{\mathcal{G}} \rangle\rangle$ denotes the set $\{\langle\langle \tilde{G} \rangle\rangle \mid \tilde{G} \in \tilde{\mathcal{G}}\}$.

Definition 3.2.6. Let $\tilde{G}_i$ be a normal formed labelled graph for $i = 1, 2$. A function $h \colon V^{\tilde{G}_2} \to V^{\tilde{G}_1}$ is called a *homomorphism* from $\tilde{G}_2$ to $\tilde{G}_1$ if the following are satisfied: (a) h is a homomorphism from $M^{\tilde{G}_2}$ to $M^{\tilde{G}_1}$; (b) $L(\tilde{G}_2) \subseteq L(\tilde{G}_1)$; (c) for any label $l \in L(\tilde{G}_2)$, $h(m^{\tilde{G}_2}(l)) = m^{\tilde{G}_1}(l)$; (d) $h(v_t^{\tilde{G}_2}) = v_t^{\tilde{G}_1}$.

$\tilde{G}_1 \lhd \tilde{G}_2$ denotes that, if $\langle\langle \tilde{G}_1 \rangle\rangle$ is defined, $\langle\langle \tilde{G}_2 \rangle\rangle$ is also defined and there exists a homomorphism from $\langle\langle \tilde{G}_2 \rangle\rangle$ to $\langle\langle \tilde{G}_1 \rangle\rangle$. $\tilde{\mathcal{G}}_1 \lhd \tilde{\mathcal{G}}_2$ denotes that, for any labelled graph $\tilde{G}_1 \in \tilde{\mathcal{G}}_1$, there exists a labelled graph $\tilde{G}_2 \in \tilde{\mathcal{G}}_2$ such that $\tilde{G}_1 \lhd \tilde{G}_2$.

Now we show a relationship between relation inclusions and homomorphisms of labelled graphs (Theorem 3.2.9).

Proposition 3.2.7 (cf. [1, Lem. 2])**.**

(1) $(M, v_s, v_t) \lhd G_1 \cdot G_2 \Leftrightarrow \exists v.(M, v_s, v) \lhd G_1 \wedge (M, v, v_t) \lhd G_2$.

(2) $(M, v_s, v_t) \lhd G_1 \parallel G_2 \Leftrightarrow (M, v_s, v_t) \lhd G_1 \wedge (M, v_s, v_t) \lhd G_2$.

(3) $(\tilde{M}, v_t) \lhd \tilde{G}_1 \cdot G_2 \Leftrightarrow \exists v.(\tilde{M}, v) \lhd \tilde{G}_1 \wedge (M^{\tilde{M}}, v, v_t) \lhd G_2$.

(4) $(\tilde{M}, v_t) \lhd \tilde{G}_1 \parallel \tilde{G}_2 \Leftrightarrow (\tilde{M}, v_t) \lhd \tilde{G}_1 \wedge (\tilde{M}, v_t) \lhd \tilde{G}_2$.

Sketch. (1) and (2) are proved in [1, Lem.2]. (3) and (4) are proved in the same way as the proof of (1) and (2). □

Proposition 3.2.8 (cf. [1, Lem. 3])**.**

(1) $(v_s, v_t) \in R_t^M \iff (M, v_s, v_t) \lhd \mathcal{G}(t)$.

(2) $v_t \in \mathrm{cod}(R_{\tilde{t}}^{\tilde{M}}) \iff (\tilde{M}, v_t) \lhd \tilde{\mathcal{G}}(\tilde{t})$.

Sketch. (1) is proved in [1, Lem. 3]. (2) is proved by induction on the size of the labelled term $\tilde{t}$ using (1), and (3) and (4) of Proposition 3.2.7. □

Theorem 3.2.9 (cf. Theorem 3.1.7)**.**

$$\mathrm{REL}^{\sim} \models \tilde{t}_1 \leq \tilde{t}_2 \iff \tilde{\mathcal{G}}(\tilde{t}_1) \triangleleft \tilde{\mathcal{G}}(\tilde{t}_2).$$

Proof. ($\Rightarrow$): Let $\tilde{G}$ be any labelled graph in $\tilde{\mathcal{G}}(\tilde{t}_1)$ such that $\langle\!\langle \tilde{G} \rangle\!\rangle$ is defined. By $(\tilde{M}^{\langle\!\langle \tilde{G} \rangle\!\rangle}, v_t^{\langle\!\langle \tilde{G} \rangle\!\rangle}) \triangleleft \tilde{G}$ ($(\tilde{M}^{\langle\!\langle \tilde{G} \rangle\!\rangle}, v_t^{\langle\!\langle \tilde{G} \rangle\!\rangle})$ is equivalent to $\langle\!\langle \tilde{G} \rangle\!\rangle$) and Proposition 3.2.8, $v_t^{\langle\!\langle \tilde{G} \rangle\!\rangle} \in \mathrm{cod}(R_{\tilde{t}_1}^{\tilde{M}^{\langle\!\langle \tilde{G} \rangle\!\rangle}})$. By $\mathrm{REL}^{\sim} \models \tilde{t}_1 \leq \tilde{t}_2$, $v_t^{\langle\!\langle \tilde{G} \rangle\!\rangle} \in \mathrm{cod}(R_{\tilde{t}_2}^{\tilde{M}^{\langle\!\langle \tilde{G} \rangle\!\rangle}})$. By Proposition 3.2.8, $(\tilde{M}^{\langle\!\langle \tilde{G} \rangle\!\rangle}, v_t^{\langle\!\langle \tilde{G} \rangle\!\rangle}) \triangleleft \tilde{\mathcal{G}}(\tilde{t}_2)$. Therefore $\tilde{G} \triangleleft \tilde{\mathcal{G}}(\tilde{t}_2)$, and thus $\tilde{\mathcal{G}}(\tilde{t}_1) \triangleleft \tilde{\mathcal{G}}(\tilde{t}_2)$. ($\Leftarrow$): Let $\tilde{M}$ be any labelled relation model and let v_t be any node in $\mathrm{cod}(R_{\tilde{t}_1}^{\tilde{M}})$. By Proposition 3.2.8, $(\tilde{M}, v_t) \triangleleft \tilde{\mathcal{G}}(\tilde{t}_1)$. By $\tilde{\mathcal{G}}(\tilde{t}_1) \triangleleft \tilde{\mathcal{G}}(\tilde{t}_2)$ and that $\triangleleft$ is transitive, $(\tilde{M}, v_t) \triangleleft \tilde{\mathcal{G}}(\tilde{t}_2)$. By Proposition 3.2.8, $v_t \in \mathrm{cod}(R_{\tilde{t}_2}^{\tilde{M}})$. Therefore $\mathrm{REL}^{\sim} \models \tilde{t}_1 \leq \tilde{t}_2$. □

Definition 3.2.10. The labelled graph of $\tilde{G}$ with a target label l_t, written $\tilde{G}^{l_t}$, is defined as $(V^{\tilde{G}}, \{R_{\hat{a}}^{\tilde{G}}\}_{\hat{a} \in \dot{A}_1}, m^{\tilde{G}} \cup \{(l_t, v_t)\}, v_t)$. $\tilde{\mathcal{G}}^{l_t}$ denotes $\{\tilde{G}^{l_t} \mid \tilde{G} \in \tilde{\mathcal{G}}\}$.

$\tilde{G}^{l_t}$ may be undefined since $m^{\tilde{G}} \cup \{(l_t, v_t)\}$ may not be functional.

Theorem 3.2.11. $\tilde{\mathcal{G}}(\tilde{t}_1) \triangleleft \tilde{\mathcal{G}}(\tilde{t}_2) \iff \forall l_t \in L. \tilde{\mathcal{G}}(\tilde{t}_1)^{l_t} \triangleleft \tilde{\mathcal{G}}(\tilde{t}_2)^{l_t}$.

Proof. ($\Rightarrow$): Let $\tilde{G}_1$ be any graph such that $\tilde{G}_1^{l_t} \in \tilde{\mathcal{G}}(\tilde{t}_1)^{l_t}$ is defined, and let $\tilde{G}_2$ be a graph such that $\tilde{G}_1 \triangleleft \tilde{G}_2$. Then, $\tilde{G}_2^{l_t}$ is defined and thus $\tilde{G}_1^{l_t} \triangleleft \tilde{G}_2^{l_t}$ by using the same homomorphism. ($\Leftarrow$): Let $\tilde{G}_1$ be any labelled graph in $\tilde{\mathcal{G}}(\tilde{t}_1)$. When $v_t^{\tilde{G}_1}$ is labelled with a label l, there is a labelled graph $\tilde{G}_2 \in \tilde{\mathcal{G}}(\tilde{t}_2)$ such that $\tilde{G}_1 \triangleleft \tilde{G}_2^l$, by $\tilde{G}_1 \in \tilde{\mathcal{G}}(\tilde{t}_1)^l$ and $\tilde{\mathcal{G}}(\tilde{t}_1)^l \triangleleft \tilde{\mathcal{G}}(\tilde{t}_2)^l$. Then $\tilde{G}_1 \triangleleft \tilde{G}_2$ by using the same homomorphism. Otherwise, let l be a label not in $L(\tilde{t}_1)$ nor $L(\tilde{t}_2)$ and let $\tilde{G}_2^l$ be a labelled graph such that $\tilde{G}_1^l \triangleleft \tilde{G}_2^l$. Then $\tilde{G}_1 \triangleleft \tilde{G}_2$ by using the same homomorphism. □

3.3 Sequential Graph Construction Procedures

In this section, we define *Sequential Graph Construction Procedures*. In a nutshell, the purpose of the procedures is to express graphs by strings. A Sequential Graph Construction Procedure (SGCP) is a string of the following events: **(Adding)**, **(Connecting)**, or **(Forgetting)**. The intuition of each event is as follows:

(Adding: $\mathtt{A}(l_1)$) Add a node labelled with l_1.

(Connecting: $\mathtt{C}(l_1, l_2, a)$) Add an edge labelled with a from the node labelled with l_1 to the node labelled with l_2.

(Forgetting: $\mathtt{F}(l_1)$) Remove the label l_1.

We use e to denote an *event* ($\mathtt{A}(l_1)$, $\mathtt{C}(l_1, l_2, a)$, or $\mathtt{F}(l_1)$) and use ρ to denote a *string of events*.

Definition 3.3.1. A *Sequential Graph Construction Procedure* (SGCP) ρ is defined as follows: (1) $\mathtt{A}(l_1)$ is a SGCP; (2) if ρ is a SGCP and $l_1 \notin \lambda(\rho)$, then $\rho\mathtt{A}(l_1)$ is a SGCP; (3) if ρ is a SGCP and $l_1, l_2 \in \lambda(\rho)$, then $\rho\mathtt{C}(l_1, l_2, a)$ is a SGCP; (4) if ρ is a SGCP and $l_1 \in \lambda(\rho)$, then $\rho\mathtt{F}(l_1)$ is a SGCP, where the *set of active labels* of a SGCP ρ, written $\lambda(\rho)$, is defined inductively by: (a) $\lambda(\mathtt{A}(l_1)) = \{l_1\}$; (b) $\lambda(\rho\mathtt{A}(l_1)) = \lambda(\rho) \cup \{l_1\}$; (c) $\lambda(\rho\mathtt{C}(l_1, l_2, a)) = \lambda(\rho)$; (d) $\lambda(\rho\mathtt{F}(l_1)) = \lambda(\rho) \setminus \{l_1\}$.

The *set of labels* of ρ, written $L(\rho)$, is defined as the set of all labels occurring in ρ. $\rho[l'/l]$ denotes ρ in which each occurrence of l has been replaced by l' and $\rho[\rho''/\rho']$ denotes ρ in which each occurrence of ρ' has been replaced by ρ''. Note that any SGCP ρ such that $\lambda(\rho) = \emptyset$ is in form of $\mathtt{A}(l)\rho'\mathtt{F}(l')$.

We explain the two main reasons of introducing labels.

(1) Considering strings as relations, any string can be regarded as a path model. Any path graph can be constructed by repeating adding an edge at target node from the singleton graph. On the above procedure, given a path graph, it is enough to know which node is target to continue the procedure. Then we can assume that the other nodes are invisible. For that reason, we do not need to use labels for path models because the number of nodes that are needed to consider continuing the procedure is unique.

However, in general, given a procedure, the number of nodes that are needed to continue the procedure may not be unique, e.g., procedures constructing tree graphs. One of the reasons to introduce labels is to distinguishes multiple nodes in procedures.

(2) Considering decidability or computational complexity, it is important to get an upperbound for the space complexity. SGCPs can make graphs with many

nodes in low space complexity under some condition related to pathwidth [74] by forgetting unnecessary labels using **Forgetting**. That is the second reason to introduce labels. In fact, any graph of a term of Kleene Allegories can be constructed in low space complexity (Theorem 3.3.10).

Definition 3.3.2. The *labelled graph of an SGCP* ρ, written $\tilde{G}(\rho) = (V^{\rho}, \{R^{\rho}_{\dot{a}}\}_{\dot{a}\in\dot{A}_1}, m^{\rho}, v^{\rho}_t)$, is a normal formed labelled graph, defined inductively as follows:

(1) if $\rho = \mathtt{A}(l_1)$, $\tilde{G}(\rho) := (\{0\}, \{\emptyset\}_{\dot{a}\in\dot{A}_1}, \{(l_1, 0)\}, 0)$;

(2) if $\rho = \rho'\mathtt{A}(l_1)$, $\tilde{G}(\rho) := (V^{\rho'} \cup \{\#V^{\rho'}\}, \{R^{\rho'}_{\dot{a}}\}_{\dot{a}\in\dot{A}_1}, m^{\rho'} \cup \{(l_1, \#V^{\rho'})\}, 0)$;

(3) if $\rho = \rho'\mathtt{C}(l_1, l_2, a)$, $\tilde{G}(\rho) := (V^{\rho'}, \{R_{\dot{a}}\}_{\dot{a}\in\dot{A}_1}, m^{\rho'}, 0)$, where $R_{\dot{a}} = R^{\rho'}_{\dot{a}} \cup \{(m^{\rho'}(l_1), m^{\rho'}(l_2))\}$ if $\dot{a} = a$, and $R_{\dot{a}} = R^{\rho'}_{\dot{a}}$ otherwise;

(4) if $\rho = \rho'\mathtt{F}(l_1)$,
$\tilde{G}(\rho) := (V^{\rho'}, \{R^{\rho'}_{\dot{a}}\}_{\dot{a}\in\dot{A}_1}, m^{\rho'} \restriction (\mathrm{dom}(m^{\rho'}) \setminus \{l_1\}), 0)$.

Note that nodes in $\tilde{G}(\rho)$ are named by natural numbers $(0, 1, \dots)$ and 0 is always used for the target node. See Figure 3.1, for a graphical description of Definition 3.3.2.

Definition 3.3.3. The *graph of a SGCP* ρ such that $\rho = \rho'\mathtt{F}(l_s)$ and $\lambda(\rho) = \emptyset$, written $G(\rho)$, is a normal formed graph, defined as $(V^{\tilde{G}(\rho')}, \{R^{\tilde{G}(\rho')}_{\dot{a}}\}_{\dot{a}\in\dot{A}_1}, m^{\tilde{G}(\rho')}(l_s), v^{\tilde{G}(\rho')}_t)$.

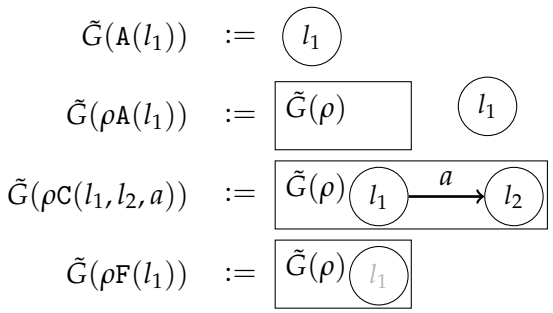


FIGURE 3.1: Inductive construction of the labelled graph of a SGCP

The next proposition shows a completeness of SGCPs with respect to express finite relations.

Proposition 3.3.4. *For any finite normal formed labelled graph* $\tilde{G}$*, there is a SGCP* ρ *such that* $\tilde{G}(\rho)$ *is isomorphic to* $\tilde{G}$*.*

Proof. Such SGCP can be constructed as follows: (a) make up nodes as many as the nodes in $\tilde{G}$ by repeating **Adding**; (b) make the same relation as $\tilde{G}$ by repeating **Connecting**; (c) remove labels not in $\tilde{G}$ by repeating **Forgetting**. □

Forgetting may seem inessential to construct graphs. However it is useful for saving the number of used labels.

Definition 3.3.5 (n-SGCP)**.** The *labelwidth* of a SGCP $\rho = e_1 \dots e_n$, written $\tilde{\#}(\rho)$, is defined as $\max\{\#\lambda(e_1 \dots e_k) \mid 1 \leq k \leq n\}$. A SGCP ρ is called an *n-SGCP* if the labelwidth of ρ is less than or equal to n.

Example 3.3.6. We consider a few examples about SGCPs.

- Let ρ_1 be $\mathtt{A}(l_n)\mathtt{A}(l_{n-1})\mathtt{C}(l_{n-1}, l_n, a_n)\mathtt{F}(l_n) \dots \mathtt{A}(l_0)$-$\mathtt{C}(l_0, l_1, a_1)\mathtt{F}(l_1)$. Then ρ_1 is a 2-SGCP and $\tilde{G}(\rho_1)$ is a path graph.
- Let ρ_2 be $\mathtt{A}(l')\mathtt{A}(l_1)\mathtt{A}(l_2)\mathtt{C}(l_1, l', a_1)\mathtt{C}(l_2, l', a_2)\mathtt{F}(l')\mathtt{A}(l)$-$\mathtt{C}(l, l_1, a_1)\mathtt{C}(l, l_2, a_2)\mathtt{F}(l_1)\mathtt{F}(l_2)\mathtt{F}(l)$.

  Then ρ_2 is a 3-SGCP and $G(\rho_2)$ is a series-parallel graph.
- Let ρ_3 be $\mathtt{A}(l_1) \dots \mathtt{A}(l_5)\mathtt{C}(l_1, l_2, a) \dots \mathtt{C}(l_5, l_4, a)$.

  Then ρ_3 is a 5-SGCP and $\tilde{G}(\rho_3)$ is the complete directed graph with 5 nodes.

See Figure 3.2 for graphs and labelled graphs expressed by the SGCPs in Example 3.3.6.

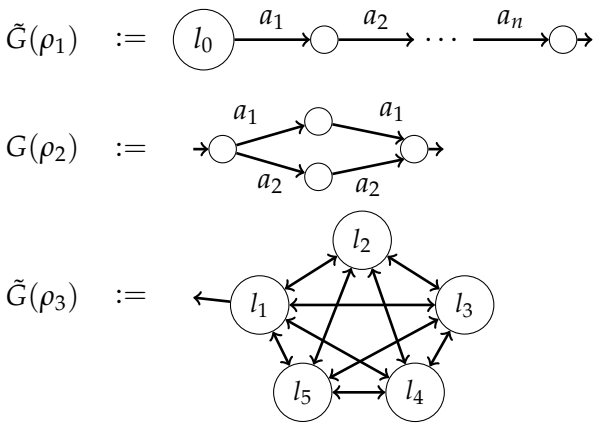


FIGURE 3.2: Examples of (labelled) graphs constructed by a SGCP

A labelled relation model $\tilde{G}$ is called *n-expressible* if there exists an n-SGCP ρ such that $\tilde{G}(\rho)$ and $\tilde{G}$ are isomorphic.

Intuitively, the smaller n that a labelled graph is n-expressible, the more the graph is "elongated" like path graphs. Theorem 3.3.10 shows that any labelled graph in the graph language of each term of Kleene allegories is "elongated".

We call that SGCPs $\rho_1, \ldots, \rho_n$ (maybe $n = 1$) are *disjoint* if, each event $\mathtt{A}(l)$ and $\mathtt{F}(l)$ occur in the SGCPs at most once.

The *series-composition* of two disjoint SGCPs, ρ_1 and ρ_2, written $\rho_1 \diamond \rho_2$, is defined as follows:

- if $\lambda(\rho_2) = \emptyset$, $\rho_1 \diamond \rho_2 := (\rho_2[\varepsilon/\mathtt{F}(l_2^s)][l_1^t/l_2^s])(\rho_1[\varepsilon/\mathtt{A}(l_1^t)])$, where $\rho_1 = \mathtt{A}(l_1^t)\rho_1'$ and $\rho_2 = \mathtt{A}(l_2^t)\rho_2'\mathtt{F}(l_2^s)$;
- otherwise, $\rho_1 \diamond \rho_2$ is undefined.

The *parallel-composition* of two disjoint SGCPs, ρ_1 and ρ_2, written $\rho_1 \parallel \rho_2$, is defined as follows:

- if $\lambda(\rho_1) = \lambda(\rho_2) = \emptyset$, $\rho_1 \parallel \rho_2 :=$ $(\rho_1[\varepsilon/\mathtt{F}(l_1^t)][\varepsilon/\mathtt{F}(l_1^s)][l_2^t/l_1^t][l_2^s/l_1^s])(\rho_2[\varepsilon/\mathtt{A}(l_2^t)][\varepsilon/\mathtt{A}(l_2^s)])$, where $\rho_1 = \mathtt{A}(l_1^t)\rho_1'\mathtt{F}(l_1^s)$ and $\rho_2 = \mathtt{A}(l_2^t)\rho_2'\mathtt{F}(l_2^s)$;
- otherwise, $\rho_1 \parallel \rho_2 := (\rho_1[\varepsilon/\mathtt{F}(l_1^t)][l_2^t/l_1^t])(\rho_2[\varepsilon/\mathtt{A}(l_2^t)])$, where $\rho_1 = \mathtt{A}(l_1^t)\rho_1'$ and $\rho_2 = \mathtt{A}(l_2^t)\rho_2'$.

Proposition 3.3.7. *Let ρ_1 and ρ_2 be disjoint SGCPs.*

(1) $G(\rho_1 \diamond \rho_2)$ is isomorphic to $G(\rho_1) \cdot G(\rho_2)$, where $\lambda(\rho_1) = \lambda(\rho_2) = \emptyset$.

(2) $\tilde{G}(\rho_1 \diamond \rho_2)$ is isomorphic to $\tilde{G}(\rho_1) \cdot G(\rho_2)$, where $\lambda(\rho_2) = \emptyset$.

(3) $G(\rho_1 \parallel \rho_2)$ is isomorphic to $G(\rho_1) \parallel G(\rho_2)$, where $\lambda(\rho_1) = \lambda(\rho_2) = \emptyset$.

(4) $\tilde{G}(\rho_1 \parallel \rho_2)$ is isomorphic to $\langle\langle \tilde{G}(\rho_1) \parallel \tilde{G}(\rho_2) \rangle\rangle$.

Proposition 3.3.8. *Let ρ_1 and ρ_2 be disjoint SGCPs.*

(1) $\tilde{\#}(\rho_1 \diamond \rho_2) = \max(\tilde{\#}(\rho_1), \tilde{\#}(\rho_2))$ if $\rho_1 \diamond \rho_2$ is defined.

(2) $\tilde{\#}(\rho_1 \parallel \rho_2) = \tilde{\#}(\rho_1) + \tilde{\#}(\rho_2)$.

Definition 3.3.9 (Intersection width (cf. [35])). The *intersection width* of a term (or a labelled term), written $\mathrm{iw}(t)$ (or $\mathrm{iw}(\tilde{t})$), is defined inductively as follows:

$$\begin{aligned}
\mathrm{iw}(0) := \mathrm{iw}(1) := \mathrm{iw}(a) := \mathrm{iw}(a^\smile) &:= 1;\\
\mathrm{iw}(t_1 \cup t_2) := \mathrm{iw}(t_1 \cdot t_2) &:= \max(\mathrm{iw}(t_1), \mathrm{iw}(t_2));\\
\mathrm{iw}(\tilde{t}_1 \cdot t_2) &:= \max(\mathrm{iw}(\tilde{t}_1), \mathrm{iw}(t_2));\\
\mathrm{iw}(t_1^*) := \mathrm{iw}(@l.t_1) &:= \mathrm{iw}(t_1);\\
\mathrm{iw}(t_1 \cap t_2) := \mathrm{iw}(t_1) + \mathrm{iw}(t_2);\ \mathrm{iw}(\tilde{t}_1 \cap \tilde{t}_2) &:= \mathrm{iw}(\tilde{t}_1) + \mathrm{iw}(\tilde{t}_2).
\end{aligned}$$

It is easy to see that $\mathrm{iw}(t) \leq |t|$ and $\mathrm{iw}(\tilde{t}) \leq |\tilde{t}|$ hold.

Theorem 3.3.10.

(1) Any graph $G \in \langle\!\langle \mathcal{G}(t) \rangle\!\rangle$ is $(\mathrm{iw}(t)+1)$-expressible.

(2) Any labelled graph $\tilde{G} \in \langle\!\langle \tilde{\mathcal{G}}(\tilde{t}) \rangle\!\rangle$ is $(\mathrm{iw}(\tilde{t})+1)$-expressible.

Proof. (1) is proved by induction on the size of t using Proposition 3.3.7 and 3.3.8. (2) is proved by induction on the size of $\tilde{t}$ using (1) and Proposition 3.3.7 and 3.3.8. □

Corollary 3.3.11. *Let l_t be any label. Any labelled graph $\tilde{G} \in \langle\!\langle \tilde{\mathcal{G}}(\tilde{t})^{l_t} \rangle\!\rangle$ is $(\mathrm{iw}(\tilde{t})+2)$-expressible.*

Proof. Let $\rho = A(l)\rho'$ be a disjoint $(\mathrm{iw}(\tilde{t})+1)$-SGCP such that ρ expresses a labelled graph $\tilde{G} \in \langle\!\langle \tilde{\mathcal{G}}(\tilde{t}) \rangle\!\rangle$. If $\tilde{G}^{l_t}$ is defined, $\rho[\varepsilon/\mathtt{F}(l_t)]$ is a $(\mathrm{iw}(\tilde{t})+2)$-SGCP such that expresses $\tilde{G}^{l_t}$. □

The labelwidth of a SGCP is closely related with space complexity. Corollary 3.3.11 is useful for analysing computational complexity in Section 3.5.

Remark. Note that the following are equivalent:

(1) $\tilde{G}$ is expressible by a n-SGCP ρ;

(2) $\tilde{G}$ is expressible by a SGCP ρ such that $\#L(\rho) \leq n$.

(2) $\Rightarrow$ (1) is obvious. (1) $\Rightarrow$ (2) is shown by relabelling each label in ρ adequately. Thus it is enough to prepare at most n labels to express a n-expressible graph $\tilde{G}$.

3.3.1 Labelwidth and Pathwidth

We mention that the labelwidth of a graph can be characterized by the *pathwidth* in graph theory [74]. (It has some alternative characterizations, see e.g., [12, Thm. 2] and [13, Lem. 4.6].) The pathwidth of a graph is determined by an optimal *path-decomposition*.

Definition 3.3.12 (cf. [74])**.** Let G be a normal formed graph. A sequence $X_1, \ldots, X_n$ of subsets of V^G is a *path-decomposition* if the following condition are satisfied:

(a) for any $(v, v') \in R^G_A$, there is i such that $\{v, v'\} \subseteq X_i$;

(b) for any $1 \leq i \leq i' \leq i'' \leq n$, $X_i \cap X_{i''} \subseteq X_{i'}$;

(c) $v^G_t \in X_1$; $v^G_s \in X_n$.

(Note that (c) is added from the definition in [74] for the source node and the target node.) Let the *width* of a path-decomposition $X_1, \ldots, X_n$ be the maximum of $\#X_1 - 1, \ldots, \#X_n - 1$. Then the *pathwidth* of a graph G is defined as the minimum of the widths of path-decompositions of G.

Example 3.3.13. Figure 3.3 shows examples of path-decompositions for graphs of Figure 3.2. These path-decompositions are optimal, i.e., the pathwidths are just 1, 2, and 4 shown by these path-decompositions, respectively.

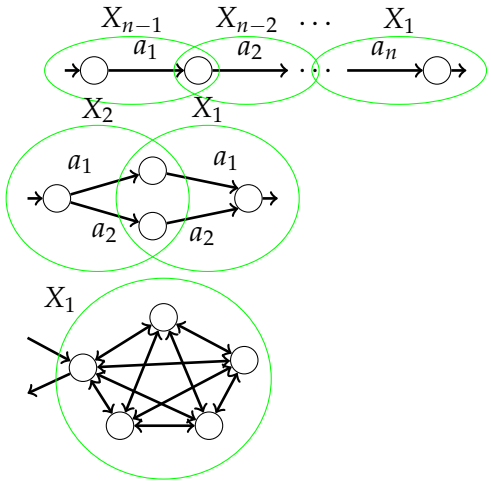


FIGURE 3.3: Examples of path-decomposition

Proposition 3.3.14. *Let $G = (V, \{R_{\dot{a}}\}_{\dot{a} \in \dot{A}_1}, v_s, v_t)$ be any normal formed graph. The following are equivalent:*

(1) G is expressible by a n-SGCP;

(2) the pathwidth of G is at most $n-1$.

Proof. (1) $\Rightarrow$ (2): Let $\rho = e_1 \dots e_k$ be a n-SGCP such that ρ expresses G, let $\rho_i = e_1 \dots e_i$ be a n-SGCP, let $\tilde{G}(\rho_i) = (V^i, \{R^i_{\dot{a}}\}_{\dot{a} \in \dot{A}_1}, m^i, 0)$ be a normal formed labelled graph, and let $X_i = \mathrm{cod}(m^i)$. Then $X_1, \dots, X_{k-1}$ is a path-decomposition of width $n-1$ of G. (Note that the condition (b) in Definition 3.3.12 is justified by that, once a label is forgotten, the node associated with the label will be never labelled again.) Thus the pathwidth of G is at most $n-1$.

(2) $\Rightarrow$ (1): Let $X_1, \dots, X_k$ be a path-decomposition of width at most $n-1$ of G. For convenience, let X_0 be $\emptyset$. We construct a SGCP according to the path-decomposition. The SGCP by repeating the next procedure from $i = 0$ to $i = k-1$ is a n-SGCP that expresses G:

(i) Forgetting any label associated with nodes in $X_i \setminus X_{i+1}$;

(ii) Adding new nodes as many as the size of $X_{i+1} \setminus X_i$;

(iii) For any pair $(v, v') \in R_a \cap ((X_{i+1} \times X_{i+1}) \setminus (X_i \times X_i))$, connecting the pair with an edge labelled with a,

where v_t is added at first when $i = 0$, and v_s is forgotten at last when $i = k-1$. □

3.4 Left Quotients on Graphs

In this section, we only consider graphs in the following class. The restriction is not critical because any labelled graph in the labelled graph language of each labelled term of Kleene allegories is simple.

Definition 3.4.1. A labelled graph $\tilde{G} = (V, \{R_{\dot{a}}\}_{\dot{a} \in \dot{A}_1}, m, v_t)$ is called *simple* if the following are satisfied:

(a) $\tilde{G}$ is acyclic: $R^+_{\dot{A}_1} \cap \Delta(V) = \emptyset$;

(b) any node is reachable to v_t: $V = \mathrm{dom}(R^*_{\dot{A}_1} \cdot \Delta(\{v_t\}))$;

(c) $\sum_{\dot{a} \in \dot{A}} \#\{v \mid (v, v') \in R_{\dot{a}}\} \leq 1$, for any node v';

(d) any left most node is labelled with some label, and vice versa: $V \setminus \mathrm{cod}(R_{\dot{A}_1}) = \mathrm{cod}(m)$.

Similarly, a graph $G = (V, \{R_{\dot{a}}\}_{\dot{a}\in\dot{A}_1}, v_s, v_t)$ is called *simple* if (a),(b),(c) and (d') $V \setminus \mathrm{cod}(R_{\dot{A}_1}) = \{v_s\}$ are satisfied.

It is easy to see that the following hold, where $X_a^{l_1,l_2}$, ρ^{-1}, $\tilde{G}\downarrow v$, and $\tilde{G}\uparrow m$ are defined in this section.

Proposition 3.4.2. *If $\tilde{G}$ is simple, the following graphs are all simple: (1) any labelled graph in $X_a^{l_1,l_2}(\tilde{G})$; (2) any labelled graph in $\rho^{-1}(\tilde{G})$; (3) $\tilde{G}\downarrow v$; (4) $\tilde{G}\uparrow m$.*

Now we define *left quotients on graphs* (generated by SGCPs). In the next section, derivatives on graphs will be defined based on the left quotients in a natural way.

Definition 3.4.3 (l-empty)**.** A labelled graph $\tilde{G}$ is said to be *l-empty* if $\tilde{M}^{\langle\!\langle \tilde{G} \rangle\!\rangle}$ is isomorphic to the labelled relation $(\{0\}, \{\emptyset\}_{\dot{a}\in\dot{A}_1}, \{(0,l)\})$.

Definition 3.4.4. The *lower labelled graph* of a labelled graph $\tilde{G}$ in $\mathfrak{G}_L$ with respect to a node $v \in V^{\tilde{G}}$, written $\tilde{G}\downarrow v$, is a labelled graph, defined as $(V', \{R_{\dot{a}}^{\tilde{G}} \cap (V' \times V')\}_{\dot{a}\in\dot{A}_1}, m^{\tilde{G}} \cap (L \times V'), v)$, where $V' = \mathrm{dom}((R_{\dot{A}_1}^{\tilde{M}})^* \cdot \Delta(\{v\}))$.

Definition 3.4.5 (Left quotients on edges)**.** The *left quotient* of a labelled graph $\tilde{G}$ with respect to a character a and two labels, l_1 and l_2, written $X_a^{l_1,l_2}(\tilde{G})$, is the smallest set of labelled graphs that the following are satisfied:

(1) Let v_1 and v_2 be two nodes such that $\tilde{G}\downarrow v_1$ is l_1-empty and $(v_1, v_2) \in R_a^{\tilde{G}}$. Then $(V^{\tilde{G}}, \{R'_{\dot{a}}\}_{\dot{a}\in\dot{A}_1}, m^{\tilde{G}} \cup \{(l_2, v_2)\}, v_t^{\tilde{G}})\downarrow v_t^{\tilde{G}} \in X_a^{l_1,l_2}(\tilde{G})$, where $R'_{\dot{a}} = R_{\dot{a}}^{\tilde{G}} \setminus \{(v_1, v_2)\}$ if $\dot{a} = a$, and $R'_{\dot{a}} = R_{\dot{a}}^{\tilde{G}}$ otherwise.

(2) Let v_1 and v_2 be two nodes such that $\tilde{G}\downarrow v_2$ is l_2-empty and $(v_2, v_1) \in R_{a^\smile}^{\tilde{G}}$. Then $(V^{\tilde{G}}, \{R'_{\dot{a}}\}_{\dot{a}\in\dot{A}_1}, m^{\tilde{G}} \cup \{(l_1, v_1)\}, v_t^{\tilde{G}})\downarrow v_t^{\tilde{G}} \in X_a^{l_1,l_2}(\tilde{G})$, where $R'_{\dot{a}} = R_{\dot{a}}^{\tilde{G}} \setminus \{(v_2, v_1)\}$ if $\dot{a} = \breve{a}$, and $R'_{\dot{a}} = R_{\dot{a}}^{\tilde{G}}$ otherwise.

$X_a^{l_1,l_2}(\tilde{\mathcal{G}})$ denotes $\bigcup_{\tilde{G}\in\tilde{\mathcal{G}}} X_a^{l_1,l_2}(\tilde{G})$.

Intuitively, $X_a^{l_1,l_2}$ means moving once from left most nodes labelled with l_1 (or l_2) to a node connected with an edge labelled with a (or $\breve{a}$), and then erasing the passed edge. Figure 3.4 is an example of left quotients on edges. The green colored edges

are the edges in which the left quotient $X_a^{l_1,l_2}$ can be triggered. The graph on the left side generates the two graphs on the right side by $X_a^{l_1,l_2}$.

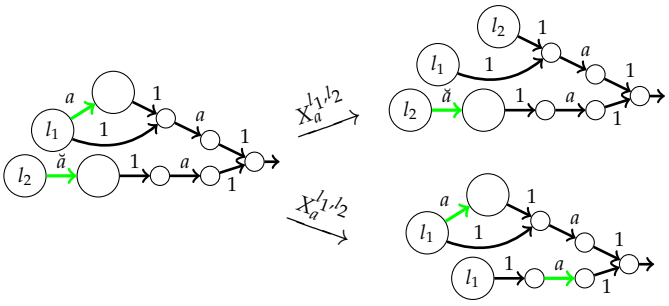


FIGURE 3.4: An example of left quotients on edges

Now we extend the left quotients from edges to graphs. Let $l^\perp$ be the *dummy label*, that is used only to define left quotients. $\tilde{\mathfrak{G}}_L^\perp$ denotes the set of labelled graphs $\tilde{\mathfrak{G}}_{L\cup\{l^\perp\}}$. $\tilde{G}^\perp$ denotes the labelled graph $(V^{\tilde{G}} \cup \{\perp\}, \{R_{\dot{a}}^{\tilde{G}}\}_{\dot{a}\in\dot{A}_1}, m_s^{\tilde{G}} \cup \{(l^\perp, \perp)\}, v_t^{\tilde{G}})$, where $\perp \notin V^{\tilde{G}}$. $m[l'/l]$ denotes m in which each pair $(l, v) \in m$ has been replaced by (l', v), and $\tilde{G}[l'/l]$ denotes $(V^{\tilde{G}}, \{R_{\dot{a}}^{\tilde{G}}\}_{\dot{a}\in\dot{A}_1}, m^{\tilde{G}}[l'/l], v_t^{\tilde{G}})$.

Definition 3.4.6 (Left quotients on graphs)**.** The *left quotient* of $\tilde{G} \in \tilde{\mathfrak{G}}^\perp$ with respect to a SGCP ρ, written $\rho^{-1}(\tilde{G})$ is a set of labelled graphs, defined inductively as follows:

$$\text{if } \rho = \mathtt{A}(l_1), \rho^{-1} := \Delta(\tilde{\mathfrak{G}}^\perp_{\{l_1\}});$$

$$\text{if } \rho = \rho'\mathtt{A}(l_1), \tilde{G}' \in \rho^{-1}(\tilde{G}) :\Leftrightarrow$$
$$\tilde{G}'[l^\perp/l_1] \in \rho'^{-1}(\tilde{G}[l^\perp/l_1]) \wedge m^{\tilde{G}'}(l_1) = m^{\tilde{G}}(l_1);$$

$$\text{if } \rho = \rho'\mathtt{C}(l_1, l_2, a), \rho^{-1} := (\rho'^{-1} \circ X_a^{l_1,l_2})^* \circ \rho'^{-1};$$

$$\text{if } \rho = \rho'\mathtt{F}(l_1), \rho^{-1} := \rho'^{-1} \cap (\tilde{\mathfrak{G}}^\perp_{\lambda(\rho)} \times \tilde{\mathfrak{G}}^\perp_{\lambda(\rho)}).$$

$\rho^{-1}(\tilde{\mathcal{G}})$ denotes $\bigcup_{\tilde{G}\in\tilde{\mathcal{G}}} \rho^{-1}(\tilde{G})$. ρ^{-1} is a closure operator.

Intuitively, ρ^{-1} means moving zero or more times between labelled nodes, and then erasing the passed edges, where each moving can be simulated by the relation model of ρ.

Example 3.4.7. Let $\rho_1 := \mathtt{A}(l_1)\mathtt{A}(l_2)\mathtt{C}(l_1, l_2, a)$. Then ρ_1^{-1} of the labelled graph on the left side of Figure 3.4 is the set of all labelled graphs obtained by applying $X_a^{l_1,l_2}$ zero or more times to the labelled graph.

Let $\rho_2 := \mathtt{A}(l_1)\mathtt{A}(l_2)\mathtt{A}(l_3)\mathtt{C}(l_3, l_2, a)\mathtt{F}(l_3)$. Then ρ_2^{-1} of the labelled graph on the left side of Figure 3.4 contains the labelled graphs by applying $X_a^{l_3,l_2}$ zero or two times, but does not contain the labelled graph by applying $X_a^{l_3,l_2}$ one time because this graph has the forgotten label l_3.

Now we show a relationship between these left quotients and homomorphisms (Theorem 3.4.11).

Lemma 3.4.8. *Let ρ be any SGCP and $\tilde{G}$ be any simple labelled graph in $\mathfrak{G}^{\perp}_{\lambda(\rho)}$.*

(1) if $(m^{\tilde{G}(\rho)^{\perp}}(l_1), m^{\tilde{G}(\rho)^{\perp}}(l_2)) \in R_a^{\tilde{G}(\rho)^{\perp}}$,
$\tilde{G}(\rho)^{\perp} \lhd X_a^{l_1,l_2}(\tilde{G}) \Longrightarrow \tilde{G}(\rho)^{\perp} \lhd \tilde{G}$.

(2) $\tilde{G}(\rho)^{\perp} \lhd \rho^{-1}(\tilde{G}) \Longrightarrow \tilde{G}(\rho)^{\perp} \lhd \tilde{G}$.

Proof. (1): Let $\tilde{G}' \in X_a^{l_1,l_2}(\tilde{G})$ be a labelled graph such that $\tilde{G}(\rho)^{\perp} \lhd \tilde{G}'$. By the definition of $X_a^{l_1,l_2}$ and that any node is reachable to $v_t^{\tilde{G}}$ (Definition 3.4.1 (b)), the subgraph with the nodes eliminated by $X_a^{l_1,l_2}$ from $\tilde{G}$ is l_1-empty (or l_2-empty). Then a homomorphism from $\langle\langle \tilde{G}' \rangle\rangle$ to $\langle\langle \tilde{G}(\rho)^{\perp} \rangle\rangle$ can be extend to a homomorphism from $\langle\langle \tilde{G} \rangle\rangle$ to $\langle\langle \tilde{G}(\rho)^{\perp} \rangle\rangle$ by mapping each eliminated node to the node labelled with l_1 (or l_2).
(2): It is proved by induction on the length of ρ.

- if $\rho = \mathtt{A}(l_1)$, it is obvious since $\rho^{-1}(\tilde{G}) = \{\tilde{G}\}$.
- if $\rho = \rho'\mathtt{A}(l_1)$, let $\tilde{G}' \in \rho^{-1}(\tilde{G})$ be a labelled graph such that $\tilde{G}(\rho)^{\perp} \lhd \tilde{G}'$. Then $\tilde{G}(\rho')^{\perp} \lhd \tilde{G}'[l^{\perp}/l_1]$ since both the node labelled with $l^{\perp}$ and the node labelled with l_1 in $\langle\langle \tilde{G}(\rho)^{\perp} \rangle\rangle$ are isolated from any other node. From this and $\tilde{G}'[l^{\perp}/l_1] \in \rho'^{-1}(\tilde{G}[l^{\perp}/l_1])$, $\tilde{G}(\rho')^{\perp} \lhd \rho'^{-1}(\tilde{G}[l^{\perp}/l_1])$. By I.H. and $\tilde{G}(\rho)^{\perp} \lhd \tilde{G}(\rho')^{\perp}$, $\tilde{G}(\rho)^{\perp} \lhd \tilde{G}[l^{\perp}/l_1]$.

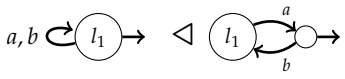
  Let h be a homomorphism from $\langle\langle \tilde{G}[l^{\perp}/l_1] \rangle\rangle$ to $\langle\langle \tilde{G}(\rho)^{\perp} \rangle\rangle$. Then a homomorphism from $\langle\langle \tilde{G} \rangle\rangle$ to $\langle\langle \tilde{G}(\rho)^{\perp} \rangle\rangle$ can be obtained from h by remapping each node labelled with l_1 in $\langle\langle \tilde{G} \rangle\rangle$ to the node labelled with l_1 in $\langle\langle \tilde{G}(\rho)^{\perp} \rangle\rangle$. Therefore, $\tilde{G}(\rho)^{\perp} \lhd \tilde{G}$.
- if $\rho = \rho'\mathtt{C}(l_1, l_2, a)$, there exists n such that $\tilde{G}(\rho)^{\perp} \lhd ((\rho'^{-1} \circ X_a^{l_1,l_2})^n \circ \rho'^{-1})(\tilde{G})$. By I.H. and $\tilde{G}(\rho)^{\perp} \lhd \tilde{G}(\rho')^{\perp}$, $\tilde{G}(\rho)^{\perp} \lhd (X_a^{l_1,l_2} \circ \rho'^{-1})^n(\tilde{G})$. By Lemma 3.4.8 (1),

$\tilde{G}(\rho)^{\perp} \triangleleft ((\rho'^{-1} \circ X_a^{l_1,l_2})^{n-1} \circ \rho'^{-1})(\tilde{G})$. By repeating the above inference, $\tilde{G}(\rho)^{\perp} \triangleleft \tilde{G}$ is proved.

- if $\rho = \rho' \mathtt{F}(l_1)$, by $\rho^{-1}(\tilde{G}) \subseteq \rho'^{-1}(\tilde{G})$, $\tilde{G}(\rho')^{\perp} \triangleleft \tilde{G}(\rho)^{\perp}$, and I.H., $\tilde{G}(\rho')^{\perp} \triangleleft \tilde{G}$. Then by using the same homomorphism, $\tilde{G}(\rho)^{\perp} \triangleleft \tilde{G}$ since $l_1 \notin L(\tilde{G})$.

□

The above lemma shows that, if a graph can be erased by a left quotient ρ^{-1}, there exists a homomorphism from the graph to $\langle\!\langle \tilde{G}(\rho) \rangle\!\rangle$. However the converse does not hold in general. For example, the below right labelled graph can not be erased by $X_a^{l_1,l_1}$, whereas there exists a homomorphism.



For that reason, the restriction to simple graphs is meaningful.

Definition 3.4.9. The *upper labelled graph* of a labelled graph $\tilde{G}$ with respect to a function m from L to $\wp(V^{\tilde{G}})$, written $\tilde{G} \uparrow m$, is a labelled graph, defined as $(V', \{R_{\dot{a}}^{\tilde{G}} \cap (V' \times V')\}_{\dot{a} \in \dot{A}_1}, (m^{\tilde{G}} \cup m) \cap (L \times V'), v_t^{\tilde{G}})$, where $V' = \mathrm{cod}((R_{\dot{A}_1}^{\tilde{M}} \cap (V \times (V \setminus \mathrm{cod}(m))))^* \cdot \Delta(v_t^{\tilde{G}}))$.

The *complement* of $\tilde{G} \uparrow m$, written $(\tilde{G} \uparrow m)^{\mathsf{c}}$, denotes the labelled graph $(V', \{R_{\dot{a}}^{\tilde{G}} \setminus R_{\dot{a}}^{\tilde{G} \uparrow m}\}_{\dot{a} \in \dot{A}_1}, (m^{\tilde{G}} \cup m) \cap (L \times V'), v_\bullet)$, where $V' = (V^{\tilde{G}} \setminus V^{\tilde{G} \uparrow m}) \cup \mathrm{cod}(m)$ and $v_\bullet$ is some node in V' ($v_\bullet$ is used only for defining the labelled graph).

Lemma 3.4.10. *Let ρ be any SGCP, let $\tilde{G}$ be any simple labelled graph in $\mathfrak{G}^{\perp}_{\lambda(\rho)}$, and let m be any function such that ($\star$): $m(l) \subseteq \mathrm{cod}(R_{\dot{A}}^{\tilde{G}})$.*

$$\tilde{M}^{\tilde{G}(\rho)^{\perp}} \triangleleft \tilde{M}^{\langle\!\langle (\tilde{G} \uparrow m)^{\mathsf{c}} \rangle\!\rangle} \Longrightarrow \tilde{G} \uparrow m \in \rho^{-1}(\tilde{G}).$$

Proof. It is proved by induction on the pair (the length of ρ, the number of edges in $\tilde{G} \uparrow m$).

- if $\rho = \mathtt{A}(l_1)$, $R_{\dot{A}}^{(\tilde{G} \uparrow m)^{\mathsf{c}}} = \emptyset$ by $R_{\dot{A}}^{\tilde{G}(\rho)^{\perp}} = \emptyset$ and the hypothesis. By ($\star$) and Definition 3.4.9, $R_{\dot{A}}^{\tilde{G} \uparrow m} = R_{\dot{A}}^{\tilde{G}}$, and thus $m = \emptyset$. Since $\tilde{G} \uparrow m = \tilde{G}$, $\tilde{G} \uparrow m \in \rho^{-1}(\tilde{G})$.

- if $\rho = \rho'\mathtt{A}(l_1)$, since both the node labelled with l_1 and the node labelled with $l^\perp$ are isolated from any other node in $\tilde{G}(\rho)^\perp$, $\tilde{M}^{\tilde{G}(\rho)^\perp} \triangleleft \tilde{M}^{\langle\!\langle(\tilde{G}[l^\perp/l_1]\uparrow m[l^\perp/l_1])^{\mathtt{c}}\rangle\!\rangle}$ by the hypothesis, and $m^{\tilde{G}\uparrow m}(l_1) = m^{\tilde{G}}(l_1)$ by $(\star)$ and Definition 3.4.9. By I.H., $\tilde{G}[l^\perp/l_1]\uparrow m[l^\perp/l_1] \in {\rho'}^{-1}(\tilde{G}[l^\perp/l_1])$, and thus $(\tilde{G}\uparrow m)[l^\perp/l_1] \in {\rho'}^{-1}(\tilde{G}[l^\perp/l_1])$. Since $m^{\tilde{G}\uparrow m}(l_1) = m^{\tilde{G}}(l_1)$, $\tilde{G}\uparrow m \in \rho^{-1}(\tilde{G})$.
- if $\rho = \rho'\mathtt{F}(l_1)$, by $\tilde{M}^{\tilde{G}(\rho')^\perp} \triangleleft \tilde{M}^{\tilde{G}(\rho)^\perp}$ and I.H., $\tilde{G}\uparrow m \in {\rho'}^{-1}(\tilde{G})$. Since $l_1 \notin L(\tilde{M}^{\tilde{G}(\rho)^\perp})$ and $\tilde{M}^{\tilde{G}(\rho)^\perp} \triangleleft \tilde{M}^{\langle\!\langle(\tilde{G}\uparrow m)^{\mathtt{c}}\rangle\!\rangle}$, $l_1 \notin L((\tilde{G}\uparrow m)^{\mathtt{c}})$. From this and $l_1 \notin L(\tilde{G})$, $l_1 \notin L(\tilde{G}\uparrow m)$. Therefore $\tilde{G}\uparrow m \in \rho^{-1}(\tilde{G})$.
- if $\rho = \rho'\mathtt{C}(l_1, l_2, a)$, Let m' be the function such that,

  $(\star 1)$ m' satisfies $(\star)$;

  $(\star 2)$ $\tilde{M}^{\tilde{G}(\rho')^\perp} \triangleleft \tilde{M}^{\langle\!\langle(\tilde{G}\uparrow m')^{\mathtt{c}}\rangle\!\rangle}$;

  $(\star 3)$ $V^{\tilde{G}\uparrow m'} \supseteq V^{\tilde{G}\uparrow m}$ is minimized.

  For short, we write $\tilde{G}\uparrow m'$ as $\tilde{G}'$. From $(\star 2)$ and I.H., $\tilde{G}' \in {\rho'}^{-1}(\tilde{G})$. When $m' = m$, it is immediately proved. Otherwise, by $m' \neq m$, $(\star 3)$, and $(\star 1)$, there exists two nodes, v_1 and v_2, such that $(v_1, v_2) \in R_{\dot{A}}^{\tilde{G}'}$ and $v_2 \in V^{(\tilde{G}\uparrow m)^{\mathtt{c}}}$. From this, the hypothesis, $(\star 2)$, and that $\tilde{G}'$ is acyclic, there exist two nodes $v_1, v_2 \in V^{\tilde{G}'}$ such that (i) $(v_1, v_2) \in R_a^{\tilde{G}'}$; (ii) $v_2 \in V^{(\tilde{G}\uparrow m)^{\mathtt{c}}}$; and (iii) $\tilde{G}'\downarrow v_1$ is l_1-empty, (or that (i) $(v_2, v_1) \in R_{\bar{a}}^{\tilde{G}'}$; (ii) $v_1 \in V^{(\tilde{G}\uparrow m)^{\mathtt{c}}}$; and (iii) $\tilde{G}'\downarrow v_2$ is l_2-empty.). Then there exists a labelled graph $\tilde{G}'' \in X_a^{l_1,l_2}(\tilde{G}')$ such that $\tilde{G}(\rho)^\perp \triangleleft \tilde{M}^{\langle\!\langle(\tilde{G}''\uparrow m)^{\mathtt{c}}\rangle\!\rangle}$. Note that $\tilde{G}''$ is simple and the number of edges in $\tilde{G}''\uparrow m$ is strictly smaller than the number in $\tilde{G}\uparrow m$. By I.H., $\tilde{G}''\uparrow m \in \rho^{-1}(\tilde{G}'')$. Since (ii), $\tilde{G}''\uparrow m = \tilde{G}'\uparrow m$. Since $(\star 3)$, $\tilde{G}'\uparrow m = \tilde{G}\uparrow m$. Also $\rho^{-1}(\tilde{G}'') \subseteq \rho^{-1}(X_a^{l_1,l_2}({\rho'}^{-1}(\tilde{G}))) \subseteq \rho^{-1}(\tilde{G})$. Therefore $\tilde{G}\uparrow m \in \rho^{-1}(\tilde{G})$.

□

Theorem 3.4.11. *Let ρ be any SGCP, let $\tilde{G}$ be any simple labelled graph in $\mathfrak{G}_{\lambda(\rho)}$, and let l_t be any label such that $\tilde{G}^{l_t}$ is defined. The following are equivalent:*

(1) there exists an l_t-empty labelled graph in $\rho^{-1}(\tilde{G})$;

(2) $\tilde{G}(\rho) \triangleleft \tilde{G}^{l_t}$.

Proof. (1) $\Rightarrow$ (2): There also exists an l_t-empty labelled graph in $\rho^{-1}(\tilde{G}^{l_t})$ because it does not matter whether the target node is labelled or not in the operation ρ^{-1}. By Lemma 3.4.8 (2), $\tilde{G}(\rho)^{\perp} \triangleleft \tilde{G}^{l_t}$. Therefore $\tilde{G}(\rho) \triangleleft \tilde{G}^{l_t}$ since $l^{\perp} \notin L(\tilde{G}^{l_t})$.
(2) $\Rightarrow$ (1): Let $m_t = \{l_t \mapsto \mathrm{cod}(R_{\dot{A}}^{\tilde{G}}) \cap ((R_1^*)^{\tilde{G}} \cdot \Delta(\{v_t^{\tilde{G}}\}))\}$. Then, since $\langle\!\langle (\tilde{G} \upharpoonright m_t)^{\mathsf{c}} \rangle\!\rangle = \langle\!\langle \tilde{G}^{l_t} \rangle\!\rangle$ and $l^{\perp} \notin L(\tilde{G})$, $M^{\tilde{G}(\rho)} \triangleleft M^{\langle\!\langle (\tilde{G} \upharpoonright m_t)^{\mathsf{c}} \rangle\!\rangle}$. By Lemma 3.4.10, $\tilde{G} \upharpoonright m_t \in \rho^{-1}(\tilde{G})$. Since $\tilde{G} \upharpoonright m_t$ is l_t-empty, (1) is proved. □

3.5 Derivatives on Graphs

In this section, we define derivatives on graphs (generated by SGCPs). The derivatives on graphs characterize the left quotient on graphs just like the derivative on strings characterize the left quotient on strings. The derivatives on graphs consist of *l-empty graph property* E_l and *partial derivatives on edges* $D_a^{l_1,l_2}$.

Definition 3.5.1 (l-empty graph property)**.** Let l be a label. The *l-empty graph property* of a labelled term, written $E_l(\tilde{t})$, is a truth value, defined inductively as follows:

$$
\begin{aligned}
E_l(@l.\dot{a}) &:= E_l(@l.0) := \mathit{false};\\
E_l(@l'.t) &:= \mathit{false} \text{ for } l' \neq l;\\
E_l(@l.1) &:= E_l(@l.t_1^*) := \mathit{true};\\
E_l(@l.t_1 \cup t_2) &:= E_l(@l.t_1) \vee E_l(@l.t_2);\\
E_l(@l.t_1 \cdot t_2) &:= E_l(@l.t_1 \cap t_2) := E_l(@l.t_1) \wedge E_l(@l.t_2);\\
E_l(\tilde{t}_1 \cdot t_2) &:= E_l(\tilde{t}_1) \wedge E_l(@l.t_2);\\
E_l(\tilde{t}_1 \cap \tilde{t}_2) &:= E_l(\tilde{t}_1) \wedge E_l(\tilde{t}_2).
\end{aligned}
$$

$E_l(\tilde{T})$ denotes the truth value $\bigvee_{\tilde{t} \in \tilde{T}} E_l(\tilde{t})$.

The next proposition shows that E_{l_t} characterizes l_t-empty. It is shown by simple induction on the size of $\tilde{t}$.

Proposition 3.5.2. *The following are equivalent:*

(1) $E_{l_t}(\tilde{t})$ is true;

(2) there exists a l_t-empty labelled graph in $\tilde{\mathcal{G}}(\tilde{t})$.

Definition 3.5.3 (Derivatives on edges)**.** The *derivatives on edges* of a labelled term $\tilde{t}$ with respect to a character a and two labels, l_1 and l_2, written $D_a^{l_1,l_2}(\tilde{t})$, is a set of labelled terms, defined inductively as follows:

$$
\begin{aligned}
D_a^{l_1,l_2}(@l.0) &:= D_a^{l_1,l_2}(@l.1) := \emptyset;\\
D_a^{l_1,l_2}(@l_1.a) &:= \{@l_2.1\};\ D_a^{l_1,l_2}(@l_2.\breve{a}) := \{@l_1.1\};\\
D_a^{l_1,l_2}(@l.a') &:= \emptyset \text{ for } a' \neq a \text{ or } l \neq l_1;\\
D_a^{l_1,l_2}(@l.\breve{a}') &:= \emptyset \text{ for } a' \neq a \text{ or } l \neq l_2;\\
D_a^{l_1,l_2}(@l.t_1 \cup t_2) &:= D_a^{l_1,l_2}(@l.t_1) \cup D_a^{l_1,l_2}(@l.t_2);\\
D_a^{l_1,l_2}(@l.t_1 \cdot t_2) &:= D_a^{l_1,l_2}((@l.t_1) \cdot t_2);\\
D_a^{l_1,l_2}(@l.t_1^*) &:= D_a^{l_1,l_2}(@l.t_1) \cdot \{t_1^*\};\\
D_a^{l_1,l_2}(@l.t_1 \cap t_2) &:= D_a^{l_1,l_2}((@l.t_1) \cap (@l.t_2));\\
D_a^{l_1,l_2}(\tilde{t}_1 \cap \tilde{t}_2) &:= (D_a^{l_1,l_2}(\tilde{t}_1) \Cap \{\tilde{t}_2\}) \cup (\{\tilde{t}_1\} \Cap D_a^{l_1,l_2}(\tilde{t}_2));
\end{aligned}
$$

$D_a^{l_1,l_2}(\tilde{t}_1 \cdot t_2)$ is the smallest set such that

(1) $D_a^{l_1,l_2}(\tilde{t}_1) \cdot \{t_2\} \subseteq D_a^{l_1,l_2}(\tilde{t}_1 \cdot t_2)$,

(2) if $E_{l_i}(\tilde{t}_1)$, $D_a^{l_1,l_2}(@l_i.t_2) \subseteq D_a^{l_1,l_2}(\tilde{t}_1 \cdot t_2)$, for $i = 1, 2$.

$D_a^{l_1,l_2}(\tilde{T})$ denotes the set of labelled terms $\bigcup_{\tilde{t}\in\tilde{T}} D_a^{l_1,l_2}(\tilde{t})$.

The above derivatives can characterize the left quotient on edges. The next proposition is shown by case analysis on where an erased edge by $X_a^{l_1,l_2}$ is derived from.

Proposition 3.5.4. *Let $\tilde{G}_1$ and $\tilde{G}_2$ be any simple labelled graphs and let G_2 be any simple graph.*

(1) $X_a^{l_1,l_2}(\tilde{G}_1 \parallel \tilde{G}_2) = (X_a^{l_1,l_2}(\tilde{G}_1) \parallel \tilde{G}_2) \cup (\tilde{G}_1 \parallel X_a^{l_1,l_2}(\tilde{G}_2))$.

(2) $X_a^{l_1,l_2}(\tilde{G}_1 \cdot G_2)$ *is the smallest set such that*

- *(a)* $X_a^{l_1,l_2}(\tilde{G}_1) \cdot \{G_2\} \subseteq X_a^{l_1,l_2}(\tilde{G}_1 \cdot G_2)$;
- *(b) if* $E_{l_i}(\tilde{G}_1)$, $X_a^{l_1,l_2}(G_2^{l_i}) \subseteq X_a^{l_1,l_2}(\tilde{G}_1 \cdot G_2)$, *for* $i = 1, 2$, *where* $\tilde{G}_2^{l_i}$ *denotes the labelled graph* $(V^{G_2}, \{R_a^{G_2}\}_{\hat{a}\in\hat{A}_1}, \{(l_i, v_s^{G_2})\}, v_t^{G_2})$.

Theorem 3.5.5. $\tilde{\mathcal{G}}(D_a^{l_1,l_2}(\tilde{t})) = X_a^{l_1,l_2}(\tilde{\mathcal{G}}(\tilde{t}))$.

Proof. It is proved by induction on the size of $\tilde{t}$ using Proposition 3.5.4. □

Next we extend the derivatives from edges to graphs.

Definition 3.5.6 (Derivatives on graphs)**.** Let e be an event and let D be a partial function from $\tilde{\mathcal{T}}$ to $\wp(\tilde{\mathcal{T}})$. Then $e(D)$ is a partial function from $\tilde{\mathcal{T}}$ to $\wp(\tilde{\mathcal{T}})$, defined as follows:

- if $e = \mathtt{A}(l_1)$ and $l_1 \notin L(D)$,
  - $e(D)(\tilde{t}_1)$ is undefined if $\tilde{t}_1 \notin \tilde{\mathcal{T}}_{L(D)\cup\{l_1\}}$,
  - $e(D)(\tilde{t}_1) := D(\tilde{t}_1)$ if $\tilde{t}_1 \in \tilde{\mathcal{T}}_{L(D)}$,
  - $e(D)(\tilde{t}_1 \cap \tilde{t}_2) := e(D)(\tilde{t}_1) \Cap e(D)(\tilde{t}_2)$,
  - $e(D)(@l_1.t) = \{@l_1.t\}$,
  - $e(D)(\tilde{t}_1 \cdot t_2) := e(D)(\tilde{t}_1) \cdot \{t_2\}$;
- if $e = \mathtt{C}(l_1, l_2, a)$ and $l_1, l_2 \in L(D)$,
  $e(D) := (D \circ D_a^{l_1,l_2})^* \circ D$;
- if $e = \mathtt{F}(l_1)$ and $l_1 \in L(D)$,
  $e(D) := D \cap (\tilde{\mathcal{T}}_{L(D)\setminus\{l_1\}} \times \tilde{\mathcal{T}}_{L(D)\setminus\{l_1\}})$;
- otherwise, $e(D)$ is undefined.

($\bullet^*$ in the definition of the case $\rho = \rho'\mathtt{C}(l_1, l_2, a)$ is the reflexive transitive closure viewed as a relation.)

The *derivative* with respect to a SGCP ρ of a labelled term $\tilde{t}$, written $D_\rho(\tilde{t})$, is defined inductively as follows:

- if $\rho = \mathtt{A}(l_1)$, $D_\rho := \{\tilde{t} \mapsto \{\tilde{t}\} \mid \tilde{t} \in \tilde{\mathcal{T}}_{\{l_1\}}\}$;
- if $\rho = \rho' e$, $D_\rho := e(D_{\rho'})$.

$D_\rho(\tilde{T})$ denotes the set of terms $\bigcup_{\tilde{t}\in\tilde{T}} D_\rho(\tilde{t})$.

Example 3.5.7. We list a few examples of derivatives on graphs.

(1) $D_{\mathtt{A}(l_0)\mathtt{A}(l_1)\mathtt{C}(l_0,l_1,a)}(@l_0.a) = \{@l_0.a, @l_1.1\}$.

(2) $D_{\mathtt{A}(l_0)\mathtt{A}(l_1)\mathtt{C}(l_0,l_1,a)\mathtt{C}(l_1,l_0,b)}(@l_0.ab \cap 1)$
$= \{@l_0.ab \cap 1, @l_1.b \cap @l_0.1, @l_0.1 \cap @l_0.1\}$.

(3) $D_{\mathtt{A}(l_0)\mathtt{A}(l_1)\mathtt{C}(l_0,l_1,a)}(@l_0.aa^\smile a)$
$= \{@l_0.aa^\smile a, @l_1.a^\smile a, @l_0.a, @l_1.1\}$.

Proposition 3.5.8. *Let $\rho = \rho'\mathtt{A}(l)$ be a SGCP. For any simple labelled graphs $\tilde{G}_1$ and $\tilde{G}_2$,*

(1) $\rho^{-1}(\tilde{G}_1 \parallel \tilde{G}_2) = \rho^{-1}(\tilde{G}_1) \parallel \rho^{-1}(\tilde{G}_2)$.

(2) if $l \in L(\tilde{G}_1)$, $\rho^{-1}(\tilde{G}_1 \cdot G_2) = \rho^{-1}(\tilde{G}_1) \cdot G_2$.

(3) if $L(\tilde{G}_1) = \{l\}$, $\rho^{-1}(\tilde{G}_1) = \{\tilde{G}_1\}$.

Sketch. These are proved by considering the range under the influence of ${\rho'}^{-1}$. □

Theorem 3.5.9. $\tilde{\mathcal{G}}(D_\rho(\tilde{t})) = \rho^{-1}(\tilde{\mathcal{G}}(\tilde{t}))$.

Sketch. It is proved by induction on the pair (the length of ρ, the size of $\tilde{t}$) using Theorem 3.5.5 for $\mathtt{C}(l_1, l_2, a)$ and using Proposition 3.5.8 for $\mathtt{A}(l_1)$. □

Theorem 3.5.10. *The following are equivalent:*

(1) $E_{l_t}(D_\rho(\tilde{t}))$;

(2) $\tilde{G}(\rho) \lhd \tilde{\mathcal{G}}(\tilde{t})^{l_t}$.

Proof. By Proposition 3.5.2 and Theorem 3.4.11 and 3.5.9. □

Summarizing the above discussion, the inclusion problem for Kleene allegories can be characterized by the derivatives on graphs as follows: $\mathrm{REL} \models t_1 \leq t_2 \Leftrightarrow \mathrm{REL}^\smile \models @l_s.t_1 \leq @l_s.t_2$ (Proposition 3.2.2) $\Leftrightarrow \forall l_t \in L.\ \tilde{\mathcal{G}}(@l_s.t_1)^{l_t} \lhd \tilde{\mathcal{G}}(@l_s.t_2)^{l_t}$ (Theorem 3.2.9 and 3.2.11) $\Leftrightarrow \forall l_t \in L. \forall \rho.\ \tilde{G}(\rho) \lhd \tilde{\mathcal{G}}(@l_s.t_1)^{l_t} \rightarrow \tilde{G}(\rho) \lhd \tilde{\mathcal{G}}(@l_s.t_2)^{l_t}$ ($\lhd$ is transitive and any labelled graph in $\tilde{\mathcal{G}}(@l_s.t_1)^{l_t}$ can be expressed by a SGCP by Corollary 3.3.11) $\Leftrightarrow \forall l_t \in L. \forall \rho.\ E_{l_t}(D_\rho(@l_s.t_1)) \rightarrow E_{l_t}(D_\rho(@l_s.t_2))$ (Theorem 3.5.10). Algorithm 2 is for deciding the above formula, where the domain of ρ is restricted to $(|t_1| + 2)$-SGCPs by Corollary 3.3.11 and cl^L is defined in the next subsection.

Algorithm 2 The inclusion problem for Kleene Allegories

Ensure: REL $\models t_1 \leq t_2$?
  Let L be a set of size $|t_1| + 2$ and let $l_s, l_t \in L$.
  $CL \Leftarrow \mathrm{cl}^L(@l_s.t_1) \cup \mathrm{cl}^L(@l_s.t_2)$
  $(D, d) \Leftarrow (\{\tilde{t} \mapsto \{\tilde{t}\} \mid \tilde{t} \in CL\}, 0)$
  while $d < 2^{\#CL^2}$ **do**
    if $E_{l_t}(D(@l_s.t_1)) \wedge \neg E_{l_t}(D(@l_s.t_2))$ **then**
      return $false$
    end if
    pickup an event e nondeterministically
    $(D, d) \Leftarrow (e(D), d + 1)$
  end while
  return $true$

3.5.1 Upper Bound

Definition 3.5.11. The *closure* of a labelled term $\tilde{t}$ with respect to a set of labels L, $\mathrm{cl}^L(\tilde{t})$, is defined inductively as follows. (For convenience, we also define $\mathrm{cl}^L(t)$ for term t.)

$\mathrm{cl}^L(0) := \bigcup_{l' \in L} \{@l'.0\}$; $\mathrm{cl}^L(1) := \bigcup_{l' \in L} \{@l'.1\}$;

$\mathrm{cl}^L(a) := \bigcup_{l' \in L} \{@l'.a, @l'.1\}$; $\mathrm{cl}^L(\breve{a}) := \bigcup_{l' \in L} \{@l'.\breve{a}, @l'.1\}$;

$\mathrm{cl}^L(t_1 \cup t_2) := \bigcup_{l' \in L} \{@l'.t_1 \cup t_2\} \cup \mathrm{cl}^L(t_1) \cup \mathrm{cl}^L(t_2)$;

$\mathrm{cl}^L(t_1 \cdot t_2) := \bigcup_{l' \in L} \{@l'.t_1 \cdot t_2\} \cup \mathrm{cl}^L(t_1) \cdot \{t_2\} \cup \mathrm{cl}^L(t_2)$;

$\mathrm{cl}^L(t_1^*) := \bigcup_{l' \in L} \{@l'.t_1^*\} \cup \mathrm{cl}^L(t_1) \cdot \{t_1^*\}$;

$\mathrm{cl}^L(t_1 \cap t_2) := \bigcup_{l' \in L} \{@l'.t_1 \cap t_2\} \cup (\mathrm{cl}^L(\tilde{t}_1) \mathbin{⫛} \mathrm{cl}^L(\tilde{t}_2))$;

$\mathrm{cl}^L(@l.t_1) := \mathrm{cl}^L(t_1)$; $\mathrm{cl}^L(\tilde{t}_1 \cdot t_2) := \mathrm{cl}^L(\tilde{t}_1) \cdot \{t_2\} \cup \mathrm{cl}^L(t_2)$;

$\mathrm{cl}^L(\tilde{t}_1 \cap \tilde{t}_2) := \mathrm{cl}^L(\tilde{t}_1) \mathbin{⫛} \mathrm{cl}^L(\tilde{t}_2)$.

$\mathrm{cl}^L(\tilde{T})$ denotes $\bigcup_{\tilde{t} \in \tilde{T}} \mathrm{cl}^L(\tilde{t})$ and cl^L is a closure operator.

Theorem 3.5.12.

(1) $D_a^{l_1, l_2}(\tilde{t}) \subseteq \mathrm{cl}^L(\tilde{t})$, *where* $\{l_1, l_2\} \cup L(\tilde{t}) \subseteq L$.

(2) $D_\rho(\tilde{t}) \subseteq \mathrm{cl}^L(\tilde{t})$, *where* $L(\rho) \cup L(\tilde{t}) \subseteq L$.

(3) $\#\,\mathrm{cl}^L(t) \leq (2 \times |t| \times \#L)^{\mathrm{iw}(t)}$.

(4) $\#\,\mathrm{cl}^L(\tilde{t}) \leq (2 \times |\tilde{t}| \times \#L)^{\mathrm{iw}(\tilde{t})}$.

Proof. (1) is proved by induction on the length of $\tilde{t}$. (2) is proved by induction on ρ using (1).

(3) is proved by induction on the length of t. (4) is proved by induction on the length of $\tilde{t}$ using (3). For example, when $\tilde{t} = \tilde{t}_1 \cap \tilde{t}_2$, it is proved as follows:

$$\begin{aligned}
\#\,\mathrm{cl}^L(\tilde{t}) &\leq \#\,\mathrm{cl}^L(\tilde{t}_1) \times \#\,\mathrm{cl}^L(\tilde{t}_2) \\
&\leq (2 \times |\tilde{t}_1| \times \#L)^{\mathrm{iw}(\tilde{t}_1)} \times (2 \times |\tilde{t}_2| \times \#L)^{\mathrm{iw}(\tilde{t}_2)} \\
&\leq (2 \times |\tilde{t}| \times \#L)^{\mathrm{iw}(\tilde{t}_1)} \times (2 \times |\tilde{t}| \times \#L)^{\mathrm{iw}(\tilde{t}_2)} \\
&\leq (2 \times |\tilde{t}| \times \#L)^{\mathrm{iw}(\tilde{t}_1)+\mathrm{iw}(\tilde{t}_2)} \\
&\leq (2 \times |\tilde{t}| \times \#L)^{\mathrm{iw}(\tilde{t})}
\end{aligned}$$

□

By $\#L \leq |\tilde{t}_1| + 2$, $\mathrm{iw}(\tilde{t}) \leq |\tilde{t}|$, and Theorem 3.5.12, $\#CL$ is an exponential of $(|t_1| + |t_2|)$. Therefore, the space complexity of Algorithm 2 is an exponential of the sum of the length of t_1 and t_2. Thus, the next theorem is derived in the same way as Theorem 2.3.26.

Theorem 3.5.13. *The inclusion problem for Kleene allegories is in* EXPSPACE.

3.5.2 A Finite Automaton Construction

The derivatives on graphs derive an algorithm turning labelled terms into finite automata like that derivatives on strings derive algorithm turning regular expressions into finite automata.

A deterministic finite automaton (DFA) $\mathcal{A}$ is a five tuple (Q, A, δ, q_I, F), where Q is a set of states, A is an alphabet, $\delta : Q \times A \to Q$ is a transition function, $q_I \in Q$ is a initial state, and $F \subseteq Q$ is a set of accepting states. For a string s, $\delta(q, s)$ denotes that, $\delta(q, s) = \{q\}$ if $s = \varepsilon$, and $\delta(q, s) = \delta(\delta(q, s'), a)$ if $s = s'a$. we call that $\mathcal{A}$ accepts a string s if $\delta(q_I, s) \in F$.

Definition 3.5.14 (DFA of a labelled term)**.** Let l_t be a label and let L be a set of labels such that $\{l_t\} \cup L(\tilde{t}) \subseteq L$. Then the DFA of a labelled term $\mathcal{A}(\tilde{t}, L, l_t) = (Q, A, \delta, q_I, F)$ is defined as follows: (a) $Q = \wp(\mathrm{cl}^L(\tilde{t}))^{\mathrm{cl}^L(\tilde{t})} \cup \{\bot\}$; (b) $A = \{\mathtt{A}(l_1) \mid l_1 \in L\} \cup \{\mathtt{C}(l_1, l_2, a) \mid l_1, l_2 \in L \text{ and } a \in A\} \cup \{\mathtt{F}(l_1) \mid l_1 \in L\}$; (c) $\delta(D, e) = e(D) \restriction \mathrm{cl}^L(\tilde{t})$ if

$e(D)$ is defined, and $\delta(D, e) = \bot$ otherwise; (d) $q_I = \Delta(\tilde{\mathcal{T}}_{\{l_t\}}) \restriction \mathrm{cl}^L(\tilde{t})$; (e) $F = \{D \mid E_{l_t}(D(\tilde{t}))$ is $true\}$.

Proposition 3.5.15. *Let ρ be any SGCP in the form of $\mathtt{A}(l_t)\rho'$.*

$$\delta^{\mathcal{A}(\tilde{t},L,l_t)}(q_I^{\mathcal{A}(\tilde{t},L,l_t)}, \rho') = D_\rho \restriction \mathrm{cl}^L(\tilde{t}).$$

Proof. It is proved by induction on the length of ρ using Theorem 3.5.12 (2). □

Theorem 3.5.16. *Let ρ be any SGCP in the form of $\mathtt{A}(l_t)\rho'$. The following are equivalent:*

(1) $E_{l_t}(D_\rho(\tilde{t}))$ is true;

(2) $\mathcal{A}(\tilde{t}, L, l_t)$ accepts ρ'.

Remark. The automata can be regarded as an extension of the automata for Kleene algebra with converse (Kleene allegories without intersection) in [11, Thm. 5.14] from path graphs to general graphs, when G in [11] is regarded as a path graph.

3.6 Lower Bound: Language and Relations

In the previous section, it is shown that the inclusion problem for Kleene allegories is in EXPSPACE. In this section, we show that the inclusion problem for Kleene allegories is EXPSPACE-complete. In the language model, a lowerbound of the universality problem is shown by the next theorem.

Theorem 3.6.1 ([33, Thm. 2])**.** *The language universality problem for terms without $\bullet^\smile$ is* EXPSPACE*-complete.*

Also the next proposition shows that the equational theory over relation models is smaller than the equational theory on the language model.

Theorem 3.6.2 (cf. [2])**.** *Let t_1 and t_2 be any terms without $\bullet^\smile$.*

$$\mathrm{REL} \models t_1 \leq t_2 \Longrightarrow \mathcal{L}(t_1) \subseteq \mathcal{L}(t_2).$$

We now show the next theorem, that is a bit stronger claim of Theorem 2.3.14.

Theorem 3.6.3. *Let t_1 and t_2 be any terms without $\bullet^\smile$ such that $\cap$ does not occur in t_1.*

$$\mathrm{REL} \models t_1 \leq t_2 \iff \mathcal{L}(t_1) \subseteq \mathcal{L}(t_2).$$

Definition 3.6.4 (Tree unwound model (see e.g., [39, p.132]))**.** The *tree unwound model* of a relation model $M = (V, \{R_a\}_{a \in A})$, written M^t, is the relation model $(V \cdot (A \cdot V)^*, \{\{(\mathbf{v}v, \mathbf{v}vav') \mid (v, v') \in R_a\}\}_{a \in A})$.

The string expressed by $\mathbf{v}$, written $s(\mathbf{v})$, is inductively defined as follows: $s(\mathbf{v}) = \varepsilon$ if $\mathbf{v} = \varepsilon$, $s(\mathbf{v}) = s(\mathbf{v}')a$ if $\mathbf{v} = \mathbf{v}'a$, and $s(\mathbf{v}) = s(\mathbf{v}')$ if $\mathbf{v} = \mathbf{v}'v$.

For tree unwound models, the next propositions hold.

Proposition 3.6.5. *Let t be any term without $\bullet^\smile$.*

(1) $(\mathbf{v}v, \mathbf{v}v\mathbf{v}') \in R_t^{M^t} \iff (v, v\mathbf{v}') \in R_t^{M^t}$.

(2) If $(\mathbf{v}, \mathbf{v}') \in R_t^{M^t}$, *there exists* $\mathbf{v}''$ *such that* $\mathbf{v}' = \mathbf{v}\mathbf{v}''$.

Lemma 3.6.6.

(1) Let t be any term without $\bullet^\smile$ nor $\cap$.

$$(v, v') \in R_t^M \Longrightarrow \exists \mathbf{v} \in (V \cdot A)^*.(v, \mathbf{v}v') \in R_t^{M^t}.$$

(2) Let t be any term without $\bullet^\smile$.

$$\exists \mathbf{v} \in (V \cdot A)^*.(v, \mathbf{v}v') \in R_t^{M^t} \Longrightarrow (v, v') \in R_t^M.$$

(3) Let t be any term without $\bullet^\smile$.

$$(v, \mathbf{v}) \in R_t^{M^t} \iff (v, \mathbf{v}) \in R_{A^*}^{M^t} \wedge s(\mathbf{v}) \in \mathcal{L}(t).$$

Proof. These are proved by simple induction on the length of t using Proposition 3.6.5. □

Proof of Theorem 3.6.3. ($\Rightarrow$): By Theorem 3.6.2. ($\Leftarrow$): We show the contraposition. Let M be a counter relation model and let (v, v') be a pair of nodes in $R_{t_1}^M \setminus R_{t_2}^M$. By

Lemma 3.6.6 (1) (2), there exists $\mathbf{v} \in (V \cdot A)^*$ such that $(v, \mathbf{v}v') \in R_{t_1}^{M^t}$ and $(v, \mathbf{v}v') \notin R_{t_2}^{M^t}$. By Lemma 3.6.6 (3), $(v, \mathbf{v}v') \in R_{A^*}^{M^t}$, $s(\mathbf{v}v') \in \mathcal{L}(t_1)$, and $s(\mathbf{v}v') \notin \mathcal{L}(t_2)$ hold. Therefore $s(\mathbf{v}v')$ is a counter string of $\mathcal{L}(t_1) \subseteq \mathcal{L}(t_2)$. Thus $\mathcal{L}(t_1) \not\subseteq \mathcal{L}(t_2)$. □

Corollary 3.6.7. *Let $A = \{a_1, \ldots, a_n\}$ and let t be any term without $\bullet^\smile$. The following are equivalent:*

(1) $\mathcal{L}(A^*) = \mathcal{L}(t)$;

(2) $\mathrm{REL} \models A^* \leq t$.

Proof. It is proved by letting t_1 be $(a_1 \cup \cdots \cup a_n)^*$ and letting t_2 be t in Theorem 3.6.3. Note that $\mathcal{L}(A^*) = \mathcal{L}(t)$ and $\mathcal{L}(A^*) \subseteq \mathcal{L}(t)$ are equivalent. □

Theorem 3.6.8.

(1) The universality problem for Kleene allegories is EXPSPACE*-complete.*

(2) The equational theory of Kleene allegories is EXPSPACE*-complete.*

Proof. (1): By Corollary 3.6.7 and Theorem 3.6.1, the problem is EXPSPACE-hard. From this and Theorem 3.5.13, the problem is EXPSPACE-complete. (2): By (1) and Theorem 3.5.13. □

3.7 Future Work

The derivatives on graphs allow us to obtain a finite automata construction algorithm for Kleene allegories. There are some natural directions for future work.

For instance, it would be interesting to apply the derivatives on graphs and the SGCPs to modal logics. The SGCPs in this paper would may be applicable to modal logics when the pathwidths of theirs models are restricted by a parameter. It would be desired some extensions of SGCPs related with *treewidth*. Also it would be interesting to extend Kleene allegories with the *negation operator* in (representable) relation algebra. The following are related work in regard to extend Kleene allegories with negation. (a) The undecidability of Kleene allegories with the negation operator is an immediate consequence of the undecidability of the relation algebra that

has the operations, negation, union, and composition [85]. (b) The fragment of the relation algebra where the negation operator is only applied to characters is decidable [57]. (c) PDL (Propositional Dynamic Logic) with intersection and negation of characters is undecidable [35]. To the best our knowledge it is open whether Kleene allegories with negation of characters is decidable.

Chapter 4

Kleene Algebra under Weak Equivalences

(This chapter is based on the author's paper [69], but some results not written in [69] are included.)

In this chapter we introduce *p-equivalence* for Kleene algebra in language semantics, that is a weak almost-equivalence based on the asymptotic probability (also called density) in formal language theory. We call that the equivalence problem (resp. universality problem, emptiness problem) under p-equivalence is p-equivalence problem (resp. p-universality problem, p-emptiness problem). For models for regular languages (terms of Kleene algebra (REGs), nondeterministic finite automata (NFAs), and deterministic finite automata (DFAs)), we show the following:

(1) For upper bound, we give a general algorithm on some equivalences characterized by logical formulas relying on descriptive complexity theory. In particular we show that the p-equivalence problem for REGs is PSPACE-complete (more precisely, NLINSPACE-complete).

(2) For lower bound, we show that the equivalence problem under every weak equivalence subsuming p-equivalence is computationally hard as the equivalence problem under the normal equivalence. Moreover we give a translation from every universality problem under the normal equivalence to an universality problem (resp. an emptiness problem) under the p-equivalence. As a corollary, the p-universality problem (resp. the p-emptiness problem)

for context-free grammars is undecidable, whereas the emptiness problem for context-free grammars is decidable (see e.g., [42, Thm. 4.1]).

In recent years some *weak (almost) equivalences* are introduced. The following are examples of weak-equivalences.

(1) $\mathcal{L}_1$ and $\mathcal{L}_2$ are f-*equivalent* [4, 5] if $\mathcal{L}_1 \triangle \mathcal{L}_2$ is finite.

(2) $\mathcal{L}_1$ and $\mathcal{L}_2$ are $\mathcal{L}$-*equivalent* ("E-equivalent" in [40]) if $\mathcal{L}_1 \triangle \mathcal{L}_2 \subseteq \mathcal{L}$, where $\mathcal{L}$ is a language.

For the equivalence problem for NFAs, the problem under the f-equivalence is PSPACE-complete and the problem under the $\mathcal{L}$-equivalence is PSPACE-complete, where $\mathcal{L}$ is given by a DFA ([40, Thm. 1]).

Remark. If $\mathcal{L}$ is fixed, the $\mathcal{L}$-equivalence problem for NFAs is not always PSPACE-hard. The language $\mathcal{L} = A^*$ is a simple counterexample.

In this chapter we define a weak equivalence based on the asymptotic probability, *p-equivalence*. Let $\mu_n(\mathcal{L})$ be

$$\mu_n(\mathcal{L}) = \frac{\text{the number of strings of length } n \text{ that are in } \mathcal{L}}{\text{the number of strings of length } n}.$$

That is, $\mu_n(\mathcal{L})$ is the probability that a randomly chosen string of length n is in $\mathcal{L}$. The *asymptotic probability* of $\mathcal{L}$, $\mu(\mathcal{L})$, is defined as $\mu(\mathcal{L}) = \lim_{n\to\infty} \mu_n(\mathcal{L})$. Then $\mathcal{L}_1$ and $\mathcal{L}_2$ are *p-equivalent* if $\mu(\mathcal{L}_1 \triangle \mathcal{L}_2) = 0$.

The definition of p-equivalence is based on the asymptotic probabilities in finite model theory, which are defined (for finite graphs) as follows. Let $\mathcal{M}$ be a set of finite graphs. Let $\mu_n(\mathcal{M})$ be

$$\mu_n(\mathcal{M}) = \frac{\text{the number of finite graphs with } n \text{ nodes that are in } \mathcal{M}}{\text{the number of finite graphs with } n \text{ nodes}}.$$

That is, $\mu_n(\mathcal{M})$ is the probability that a randomly chosen graph with n nodes is in $\mathcal{M}$. (Note that this definition can be naturally extended to any finite σ-structures from finite graphs.) The asymptotic probability of $\mathcal{M}$, written $\mu(\mathcal{M})$, is defined as $\mu(\mathcal{M}) = \lim_{n\to\infty} \mu_n(\mathcal{M})$. Then we define that $\mathcal{M}$ is *almost surely valid* if $\mu(\mathcal{M}) = 1$.

The class of models satisfying a formula φ is denoted by $\mathcal{M}(\varphi)$. φ is *almost surely valid* if $\mu(\mathcal{M}(\varphi)) = 1$. The following two results show that there is a decidability gap between validity and almost surely validity.

Theorem 4.0.1 (Trakhtenbrot's Theorem, see e.g., [55, Thm. 9.2])**.** *For every signature σ with at least one binary relation symbol, it is* undecidable *whether a first order sentence φ is valid over finite σ-structures.*

Theorem 4.0.2 (see e.g., [55, Cor. 12.11])**.** *Let σ be a fixed signature. Then, for every first order sentence φ, it is* decidable *whether φ is almost surely valid over finite σ-structures.*

More precisely, the problem in Theorem 4.0.2 is PSPACE-complete [36], whereas the problem in Theorem 4.0.1 is coRE-complete.

One of our main motivation to introduce p-equivalence was as follows: Is there some difference in decidability or computational complexity between the normal equivalence and p-equivalence (like the relationship between Theorem 4.0.1 and Theorem 4.0.2)? In contrast with that the almost surely validity problem is easier than the validity problem for first-order logic, this paper shows that many language problems under p-equivalence are not computationally easier than them under the normal equivalence.

The upper bound is shown by giving logical characterizations of p-equivalence and several results in descriptive complexity. In particular we show the following for DFAs.

Lemma 4.0.3. *For every DFA $\mathcal{A} = (Q, \delta, q^I, F)$ over a (nonempty finite) alphabet A, the following are equivalent:*

(1) $\mu(L(\mathcal{A})) \neq 0$;

(2) $\mathcal{A} \models (\exists q \mid F(q) \wedge \mathrm{Reach}(q^I, q)).(\forall q' \mid \mathrm{Reach}(q, q')).\,\mathrm{Reach}(q', q)$

where $\mathrm{Reach}(q, q')$ *means that there is a string s such that $\delta(q, s) = q'$.*

The lower bound cannot be shown straightforward from the computational hardness under the normal equivalence, but can be shown by modifying the proofs of the hardness under the normal equivalence. We also give a translation on languages from the universality problem to the *p*-universality problem. This translation shows

that both the p-universality problem and the p-emptiness problem for context-free grammars are undecidable.

Finally we also introduce *p-equivalence* in relational semantics in the same way. In this case every equational formula of Kleene algebra terms satisfies the zero one law because it is expressed by an $\mathsf{FO}(\mathsf{TC})$ formula. We remain open the computational complexity of language problems under the p-equivalence in relational semantics.

4.1 p-equivalence in Language Semantics

In this section we define p-equivalence in language semantics and some fundamental results of p-equivalence.

Let A be a (nonempty finite) alphabet and let $\mathcal{L}$ be a language over A. Let $\mu_n(\mathcal{L})$ be

$$\mu_n(\mathcal{L}) = \frac{\#(\mathcal{L} \cap A^n)}{\#(A^n)}$$

That is, $\mu_n(\mathcal{L})$ is the probability that a string of n length given by uniform randomly is in $\mathcal{L}$. The *asymptotic probability* of $\mathcal{L}$ as $\mu(\mathcal{L}) = \lim_{n\to\infty} \mu_n(\mathcal{L})$.

Definition 4.1.1 (p-equivalence)**.** Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be languages. We call that $\mathcal{L}_1$ and $\mathcal{L}_2$ are *p-equivalent* if $\mu(\mathcal{L}_1 \triangle \mathcal{L}_2) = 0$.

$\mathcal{L}_1 \simeq_p \mathcal{L}_2$ denotes that $\mathcal{L}_1$ and $\mathcal{L}_2$ are p-equivalent. $t_1 \simeq_p t_2$ denotes that $\mathcal{L}(t_1) \simeq_p \mathcal{L}(t_2)$, where t_1 and t_2 are regular expressions.

We give some examples of the asymptotic probability μ and the p-equivalence.

Example 4.1.2 (asymptotic probability μ).

- Obviously, $\mu(A^*) = 1$ and $\mu(\emptyset) = 0$.
- Let $A = \{a, b\}$ and let $\mathcal{L}_1$ be the language of all strings not including the character b (i.e., $\mathcal{L}_1 = L(a^*)$). Then $\mu_n(\mathcal{L}_1) = \frac{1}{2^n}$ and thus $\mu(\mathcal{L}_1) = 0$. Note that, if let $A = \{a\}$, $\mu(\mathcal{L}_1) = 1$. As this example, the asymptotic probability is depend on the alphabet A.

- Let $\mathcal{L}_1$ be the language of all strings of even length. $\mathcal{L}_1$ is regular since $\mathcal{L}_1 = \mathcal{L}((A \cdot A)^*)$. Then $\mu_n(\mathcal{L}_1) = 1$ if n is even and $\mu_n(\mathcal{L}_1) = 0$ if n is odd, and thus $\mu(\mathcal{L}_1)$ does not exist. Note that $\mu(L)$ does not always exist.

Example 4.1.3 (p-equivalence). Let $t_1 = (a \cup b)^*$, $t_2 = a \cdot (a \cup b)^*$, and $t_3 = 0$.

- Let $A = \{a, b\}$. Then $\mu_n(\mathcal{L}(t_1) \triangle \mathcal{L}(t_2)) = \frac{\#(b \cdot A^{n-1})}{\#(A^n)} = \frac{1}{2}$, and thus $\mu(\mathcal{L}(t_1) \triangle \mathcal{L}(t_2)) = \frac{1}{2}$. Therefore $t_1 \not\simeq_p t_2$.
- Let $A = \{a, b\}$. Then $\mu_n(\mathcal{L}(t_1) \triangle \mathcal{L}(t_3)) = 1$, and thus $\mu(\mathcal{L}(t_1) \triangle \mathcal{L}(t_3)) = 1$. Therefore $t_1 \not\simeq_p t_3$.
- Let $A = \{a, b, c\}$. Then $\mu_n(\mathcal{L}(t_1) \triangle \mathcal{L}(t_3)) = \frac{2^n}{3^n}$, and thus $\mu(\mathcal{L}(t_1) \triangle \mathcal{L}(t_3)) = 0$. Therefore $t_1 \simeq_p t_3$.

p-equivalence is indeed an equivalence relation.

Proposition 4.1.4. *p-equivalence satisfies the following:*

(1) reflexive: $\mathcal{L} \simeq_{\mathrm{p}} \mathcal{L}$;

(2) symmetric: $\mathcal{L}_1 \simeq_{\mathrm{p}} \mathcal{L}_2 \Longrightarrow \mathcal{L}_2 \simeq_{\mathrm{p}} \mathcal{L}_1$;

(3) transitive: $\mathcal{L}_1 \simeq_{\mathrm{p}} \mathcal{L}_2$ *and* $\mathcal{L}_2 \simeq_{\mathrm{p}} L_3 \Longrightarrow \mathcal{L}_1 \simeq_{\mathrm{p}} L_3$.

Proof. (1) and (2) are immediate from the definition of p-equivalence. For (3), $0 \leq \mu_n(\mathcal{L}_1 \triangle L_3) \leq \mu_n(\mathcal{L}_1 \triangle \mathcal{L}_2) + \mu_n(\mathcal{L}_2 \triangle L_3)$ holds. Then, by the assumption $\mu(\mathcal{L}_1 \triangle \mathcal{L}_2) = \mu(\mathcal{L}_2 \triangle L_3) = 0$ and the squeeze theorem, $\mu(\mathcal{L}_1 \triangle L_3) = 0$ holds. □

We show a relationship between p-equivalence and f-equivalence.

Proposition 4.1.5.

(1) $= \subseteq \simeq_{\mathrm{f}} \subseteq \simeq_{\mathrm{p}}$.

(2) If $\#A \geq 2$, *then* $\simeq_{\mathrm{f}} \subsetneq \simeq_{\mathrm{p}}$.

(3) If $\#A = 1$, *then* $\simeq_{\mathrm{p}}$ *is equal to* $\simeq_{\mathrm{f}}$.

Proof. (1) $\simeq_{\mathrm{f}} \subseteq \simeq_{\mathrm{p}}$ is shown by that $\mu(\mathcal{L}_1 \triangle \mathcal{L}_2) = 0$ if if $\mathcal{L}_1 \triangle \mathcal{L}_2$ is finite. (2) The second example in Example 4.1.3 is a counterexample. (3) By (1), it is enough to prove $\simeq_{\mathrm{f}} \supseteq \simeq_{\mathrm{p}}$. We prove the contraposition , i.e., if $\mathcal{L}_1 \not\simeq_{\mathrm{f}} \mathcal{L}_2$, then $\mathcal{L}_1 \not\simeq_{\mathrm{p}} \mathcal{L}_2$. Note that $\mu_n(\mathcal{L}_1 \triangle \mathcal{L}_2)$ is always 0 or 1, since $\#A^n = 1$ holds by $\#A = 1$. If

$\mathcal{L}_1 \not\approx_{\mathrm{f}} \mathcal{L}_2$, then $\mathcal{L}_1 \triangle \mathcal{L}_2$ is infinite, i.e., $\mu_n(\mathcal{L}_1 \triangle \mathcal{L}_2) = 1$ occurs infinitely. Therefore $\lim_{n\to\infty} \mu_n(\mathcal{L}_1 \triangle \mathcal{L}_2) \neq 0$. Hence $\mathcal{L}_1 \not\approx_{\mathrm{p}} \mathcal{L}_2$. □

4.1.1 A Robustness

Some other definitions of the asymptotic probability of $\mathcal{L}$ are considerable such as follows:

$$\mu_n(\mathcal{L}) = \frac{\#(\mathcal{L} \cap A^n)}{\#(A^n)}$$

$$\mu_n^*(\mathcal{L}) = \frac{\#(\mathcal{L} \cap A^{<n})}{\#(A^{<n})}$$

$$\delta_n(\mathcal{L}) = \frac{\sum_{k=0}^{n-1} \mu_k(\mathcal{L})}{n}$$

where $A^{<n} = \bigcup_{0\leq k<n} A^k$. Let $\mu^*(\mathcal{L})$ (resp. $\delta(\mathcal{L})$) be the limit of $\mu_n^*(\mathcal{L})$ (resp. $\delta_n(\mathcal{L})$) as n approaches infinitely. These three definition has been used in previous works: μ_n [7, 76, 80], μ_n^* [7], and δ_n [8] (see [80, Sec. 2.4] for more details).

In this subsection we prove that the three p-equivalences defined by μ, μ^*, or δ are all equivalent over regular languages (Proposition 4.1.8). To prove it, we introduce the following two theorems.

Theorem 4.1.6 (Stolz-Cesàro theorem, see e.g., [67, Thm. 1.22])**.** *If* $\lim_{n\to\infty} \frac{a_{n+1}-a_n}{b_{n+1}-b_n} = l$*, then* $\lim_{n\to\infty} \frac{a_n}{b_n} = l$*, where* $\{a_n\}_{n=0}^{\infty}$ *is a sequence of integers,* $\{b_n\}_{n=0}^{\infty}$ *is a sequence of integers and strictly monotone, and* l *is a real number.*

Theorem 4.1.7 ([58, Thm. 1.2])**.** *For every regular language* $\mathcal{L}$*, there is a positive integer* a *such that, for any integer* $0 \leq b < a$*,* $\lim_{n\to\infty} \mu_{an+b}(\mathcal{L}) = l_b$ *exists.*

Proposition 4.1.8. *For every regular language* $\mathcal{L}$*, the following are all equivalent:*

(1) $\mu(\mathcal{L}) = 0$*;*

(2) $\mu^*(\mathcal{L}) = 0$*;*

(3) $\delta(\mathcal{L}) = 0$*.*

Proof. (1) ⇒ (2) and (1) ⇒ (3) are directly derived from Stolz-Cesàro Theorem (Theorem 4.1.6). (These parts hold even if $\mathcal{L}$ is not a regular language.)

We now prove the converse using Theorem 4.1.7 and Theorem 4.1.6.

(3) ⇒ (1): Let us consider the following inequality.

$$\delta_n(\mathcal{L}) = \sum_{k=0}^{n-1} \frac{\mu_k(\mathcal{L})}{n} \geq \sum_{b=0}^{a-1} \sum_{k'=0}^{m-1} \frac{\mu_{ak'+b}(\mathcal{L})}{n} = \sum_{b=0}^{a-1} \frac{\sum_{k'=0}^{m-1} \mu_{ak'+b}(\mathcal{L})}{am} \times \frac{am}{n}$$

where $m = \lfloor \frac{n}{a} \rfloor$ and a is an integer in Theorem 4.1.7. Then the limit of the right-hand side as m approaches infinity is $\sum_{b=0}^{a-1} \frac{l_b}{a}$ by Theorem 4.1.6 (let $a_m = \sum_{k'=0}^{m-1} \mu_{ak'+b}(\mathcal{L})$ and $b_m = am$). By that $\lim_{n\to\infty} \delta_n(\mathcal{L}) = 0$, l_b is equal to 0 for any $0 \leq b < a$. Therefore $\lim_{n\to\infty} \mu_n(\mathcal{L}) = 0$.

(2) ⇒ (1): Let us consider the following inequality.

$$\begin{aligned}\mu^*_n(\mathcal{L}) = \sum_{k=0}^{n-1} \frac{\mu_k(\mathcal{L}) \times \#(A^k)}{\#(A^{<n})} &\geq \sum_{b=0}^{a-1} \sum_{k'=0}^{m-1} \frac{\mu_{ak'+b}(\mathcal{L}) \times \#(A^{ak'+b})}{(\#A)^{<n}} \\ &= \sum_{b=0}^{a-1} \frac{\sum_{k'=0}^{m-1} \mu_{ak'+b}(\mathcal{L}) \times \#(A^{ak'+b})}{\sum_{k'=0}^{m-1} \#(A^{ak'+b})} \times \frac{\sum_{k'=0}^{m-1} \#(A^{ak'+b})}{\#(A^{<n})}\end{aligned}$$

where $m = \lfloor \frac{n}{a} \rfloor$ and a is an integer in Theorem 4.1.7. Then the limit of the right-hand side as m approaches infinity is $\sum_{b=0}^{a-1} l_b \times \frac{\#(A^b)}{\#(A^{<a})}$ by Theorem 4.1.6 (let $a_m = \sum_{k'=0}^{m-1} \mu_{ak'+b}(\mathcal{L}) \times \#(A^{ak'+b})$ and $b_m = \sum_{k'=0}^{m-1} \#(A^{ak'+b})$). By that $\lim_{n\to\infty} \mu^*_n(\mathcal{L}) = 0$, l_b is equal to 0 for any b. Therefore $\lim_{n\to\infty} \mu_n(\mathcal{L}) = 0$. □

Remark. In Proposition 4.1.8, (3) ⇒ (1) (resp. (2) ⇒ (1)) does not hold for arbitrary languages. The language $\mathcal{L} = \{s \in A^* \mid \exists n \in \mathbb{N}.|s| = 2^n\}$ is a counterexample. Then $\mu^*(\mathcal{L}) = \delta(\mathcal{L}) = 0$, but $\mu(\mathcal{L})$ does not exist.

4.1.2 A Logical Characterization of p-equivalence

In this subsection we give a logical characterization of p-equivalence. In addition, we also give a logical characterization of f-equivalence. These characterizations are useful to give the computational upper bound of language problems under p-equivalence. As related work, Sin'ya gave some characterizations of that a regular language satisfies the zero-one law [80, Thm. 2.3.1]. The proof of Lemma 4.0.3 also gives an elementary proof of the main part of [80, Thm. 2.3.1].

Lemma 4.0.3 (restated). For every DFA $\mathcal{A} = (Q, \delta, q^I, F)$, the following are equivalent:

(1) $\mu(\mathcal{L}(\mathcal{A})) \neq 0$;

(2) $(\exists q \mid \mathrm{Reach}(q^I, q)).(F(q) \wedge (\forall q' \mid \mathrm{Reach}(q, q')).\mathrm{Reach}(q', q))$.

Intuitively, $(\forall q' \mid \mathrm{Reach}(q, q')).\mathrm{Reach}(q', q)$ in (2) means that the SCC (Strongly Connected Component [26, Sec. 22.5]) of q is a sink SCC.

Proof. Let $\mu_n(q) = \frac{\#\{s \in A^n \mid \delta(q^I, s) = q\}}{\#A^n}$, let $\mu_n(Q') = \sum_{q \in Q'} \mu_n(q)$, and let $\mu(Q')$ be the limit of $\mu_n(Q')$ as $n \to \infty$. Note that $\mu_n(L(\mathcal{A})) = \mu_n(F)$ holds.

(1) $\Rightarrow$ (2): We prove the contraposition, i.e., if $(\forall q \mid \mathrm{Reach}(q^I, q) \wedge F(q)).(\exists q' \mid \mathrm{Reach}(q, q')).\neg\mathrm{Reach}(q', q)$, then $\mu(F) = 0$. ${}_\bullet Q_q \subseteq Q$ denotes the set of all nodes that are reachable to q and ${}_q Q_\bullet \subseteq Q$ denotes the set of all nodes that are reachable from q, where q is a state in Q. ${}_q Q_{q'}$ denotes ${}_q Q_\bullet \cap {}_\bullet Q_{q'}$.

We first show the following: ($\heartsuit$) for any $q \in {}_{q^I} Q_\bullet \cap F$ and any $k \geq \#Q$, the inequality $\mu_k({}_{q^I} Q_q) \leq (1 - \frac{1}{\#A^{\#Q}}) \times \mu_{k-\#Q}({}_{q^I} Q_q)$ holds. By the assumption (i.e., $(\exists q' \mid \mathrm{Reach}(q, q')).\neg\mathrm{Reach}(q', q)$), for every $q'' \in {}_{q^I} Q_q$, there is a string s' such that $\delta(q'', s') \notin {}_{q^I} Q_q$. Then we can assume that $|s'| = \#Q$. (It is because the shortest length of string s' satisfying $\delta(q'', s') \notin {}_{q^I} Q_q$ is at most $\#Q$ and $\delta(q'', s') \notin {}_{q^I} Q_q \Longrightarrow \delta(q'', s's'') \notin {}_{q^I} Q_q$ holds for any string s''.) Therefore ($\heartsuit$) holds.

Let us come back to the proof of (1) $\Rightarrow$ (2).

$$\begin{aligned}\mu_k(F) = \sum_{q \in F \cap {}_{q^I} Q_\bullet} \mu_k(q) \leq \sum_{q \in F} \mu_k({}_{q^I} Q_q) \leq & \sum_{q \in F} (1 - \frac{1}{(\#A)^{\#Q}})^{\lfloor \frac{k}{\#Q} \rfloor} \times \mu_{(k \bmod \#Q)}({}_{q^I} Q_q) \\ & \text{(by applying } \heartsuit \text{ repeatedly)} \\ \leq & \#F \times (1 - \frac{1}{(\#A)^{\#Q}})^{\lfloor \frac{k}{\#Q} \rfloor} \times 1\end{aligned}$$

In the above inequality, the limit of the right-hand side as $k \to \infty$ is 0. Therefore $\mu(F) = 0$.

(2) $\Rightarrow$ (1): Let q be a state satisfying (2), let s_0 be a string such that $\delta(q^I, s_0) = q$, and let S_q be the SCC of q. Note that S_q is a sink SCC by the assumption (i.e., $(\forall q' \mid \mathrm{Reach}(q, q')).\mathrm{Reach}(q', q)$), and thus, $\mu_k(S_q) \geq \frac{1}{\#A^{|s_0|}}$ holds for any $k \geq |s_0|$. By the pigeon hole principle and that S_q is a sink SCC, for any $k \geq |s_0|$, there is a state $q' \in S_q$ such that $\mu_k(q') \geq \frac{\mu_k(S_q)}{\#S_q}$. Let $s_{q'}$ be a string such that $\delta(q', s_{q'}) = q$. We can assume that $|s_{q'}| \leq \#S_q$. (It is because that we can reach to q from any state $q' \in S_q$ at

most $\#S_q$ steps.) Let consider the following inequality for every $k \geq |s_0|$.

$$\mu_{k+|s_{q'}|}(F) \geq \mu_{k+|s_{q'}|}(q) \geq \mu_k(q') \times \frac{1}{(\#A)^{|s_{q'}|}} \geq \frac{\mu_k(S_q)}{\#S_q} \times \frac{1}{(\#A)^{\#S_q}}$$
$$\geq \frac{1}{(\#A)^{|s_0|}} \times \frac{1}{\#S_q} \times \frac{1}{(\#A)^{\#S_q}}$$

Then $\mu(F) = 0$ (i.e., $\forall \varepsilon > 0. \exists N. \forall n > N. |\mu_n(F)| < \varepsilon$) does not hold because $\varepsilon = \frac{1}{(\#A)^{|s_0|}} \times \frac{1}{\#(S_q)} \times \frac{1}{(\#A)^{\#(S_q)}}$ is a counter example by the above inequality. Therefore $\mu(F) \neq 0$. □

We also give a logical characterization for f-equivalence. (Let us recall that p-equivalence is equivalent to f-equivalence if $\#A = 1$.)

Lemma 4.1.9. *For every NFA $\mathcal{A} = (Q, \{\delta_a\}_{a \in A}, q^I, F)$, the following are equivalent:*

(1) $\mathcal{L}(\mathcal{A}) \not\approx_{\mathrm{f}} \emptyset$;

(2) $(\exists q \mid \mathrm{Reach}(q^I, q)).F(q) \wedge \mathrm{Reach}^+(q, q)$.

where $\mathrm{Reach}(q, q')$ *means that there is a string s such that $\delta(q, s) = q'$ and* $\mathrm{Reach}^+(q, q')$ *means that there is a non-empty string s such that $\delta(q, s) = q'$.*

Proof. (2) $\Rightarrow$ (1): Easy.

(1) $\Rightarrow$ (2): Let us consider the contraposition. Then it is easy to see that, if $(\forall q \mid \mathrm{Reach}(q^I, q) \wedge F(q)).\neg\, \mathrm{Reach}^+(q, q)$, then $\#\mathcal{L}(\mathcal{A})$ is finite. □

(Note that this characterization is also applicable to NFAs in contrast to the characterization in Lemma 4.0.3).

By Proposition 4.1.5, the formula in Lemma 4.1.9 also characterizes the p-emptiness for the unary case. Finally we remark that every p-equivalence problem over any alphabet is reducible to the p-equivalence problem over the binary alphabet.

Proposition 4.1.10. *Let A be a fixed nonempty finite alphabet. Then there is a log-space reduction from every p-equivalence problem for NFAs (resp. DFAs, REGs) over A to an p-equivalence problem for NFAs (resp. DFAs, REGs) over the alphabet $\{0, 1\}$.*

Proof Sketch. It is easily proved by encoding every character A to a binary string in the same manner as [48, Fig. 2.]. □

4.1.3 p-equivalence on Prefix-closed Languages

In this subsection we investigate *prefix-closed languages* under p-equivalence. We show that every universality problem of a language can be translated to an universality problem of a prefix-closed language (Lemma 4.1.15). Moreover, for prefix-closed languages, the p-universality problem is same as the universality problem (Proposition 4.1.12(3)). Combining these, we can translate every universality problem to a p-universality problem (Corollary 4.1.17).

Definition 4.1.11 ([20, 48])**.** A language $\mathcal{L}$ is *prefix-closed* if the following holds: for any string s in $\mathcal{L}$, every prefix string of s is also in $\mathcal{L}$.

For the asymptotic probability, the following holds.

Proposition 4.1.12. *Let $\mathcal{L}$ be a prefix-closed language over a nonempty finite alphabet A. Then,*

(1) The sequence $(\mu_0(\mathcal{L}), \mu_1(\mathcal{L}), \dots)$ is monotonically decreasing, i.e., $\mu_0(\mathcal{L}) \geq \mu_1(\mathcal{L}) \geq \cdots \geq \mu_n(\mathcal{L}) \geq \cdots$.

(2) $\mu(\mathcal{L})$ always exists.

(3) $\mathcal{L} = A^* \iff \mu(\mathcal{L}) = 1$.

Proof. (1): $A^{n+1} \cap \mathcal{L} \subseteq (A^n \cap \mathcal{L}) \cdot A$ holds by that $\mathcal{L}$ is a prefix-closed language. Therefore, $\mu_{n+1}(\mathcal{L}) = \frac{\#(A^{n+1} \cap \mathcal{L})}{\#(A)^{n+1}} \leq \frac{\#((A^n \cap \mathcal{L}) \cdot A)}{\#(A)^{n+1}} \leq \frac{\#(A^n \cap \mathcal{L})}{\#(A)^n} = \mu_n(\mathcal{L})$.

(2): By (1).

(3)$\Rightarrow$: By $= \subseteq \simeq_{\mathrm{p}}$.

(3)$\Leftarrow$: We prove the contraposition. Let s be a string not in $\mathcal{L}$. Then every string s' such that s is a prefix of s' is also not in $\mathcal{L}$ by that $\mathcal{L}$ is a prefix-closed language, and thus $sA^* \cap \mathcal{L} = \emptyset$. Therefore, by that $\mu_n(\mathcal{L}) \leq 1 - \frac{1}{(\#A)^{|s|}}$ holds for any $n \geq |s|$, $\mu(\mathcal{L}) \neq 1$. □

A Translation from Universality to p-universality

For NFAs, it is known that the universality problem is hard even if we assume that every language is prefix-closed by the following two theorems.

Theorem 4.1.13 ([48, Thm. 8.(a)]). *A nonempty regular language $\mathcal{L}$ is prefix-closed if and only if $\mathcal{L}$ is recognized by some NFA such that every state is an acceptance state.*

Theorem 4.1.14 ([48, Thm. 1] (cf. [43, Prop. 2.4])). *The universality problem for NFAs that have a prefix-closed language is* PSPACE*-hard.*

Let A be a nonempty finite alphabet. In this subsection, we assume that $A \cap \{\#\} = \emptyset$ and $A_\#$ denotes $A \cup \{\#\}$. $\mathcal{L}^\triangleleft$ denotes the *prefix-closure* of a language $\mathcal{L}$, i.e., $\mathcal{L}^\triangleleft$ is the smallest prefix-closed language subsuming $\mathcal{L}$. We define $\mathcal{L}^{\triangleleft_1}$ as follows:

$$\mathcal{L}^{\triangleleft_1} := (\mathcal{L}\#)^*\mathcal{L}^\triangleleft.$$

This construction is an analogy of [48, Fig.1.] for NFAs, but this can applies to the case of REGs and cases beyond regular languages.

Lemma 4.1.15. *Let $\mathcal{L}$ be any language over A. Then,*

(1) $\mathcal{L}^{\triangleleft_1}$ is prefix-closed;

(2) $\mathcal{L} = A^* \iff \mathcal{L}^{\triangleleft_1} = A_\#^*$.

Proof. (1): We prove the following by induction on the length of a string s ($\heartsuit$): if s is in $\mathcal{L}^{\triangleleft_1}$, then every prefix s' of s is in $\mathcal{L}^{\triangleleft_1}$. First, by the construction of $\mathcal{L}^{\triangleleft_1}$, s is denoted by $s_1\#s_2\#\ldots\#s_n\#s_{n+1}$, where $s_i \in \mathcal{L}$ for $i \in [n]$ and $s_{n+1} \in \mathcal{L}^\triangleleft$. If $s_{n+1} \neq \varepsilon$, then $s_1\#s_2\#\ldots\#s_n\#s'_{n+1} \in \mathcal{L}^{\triangleleft_1}$ by $s'_{n+1} \in \mathcal{L}^\triangleleft$, where s'_{n+1} is the prefix of s_{n+1} of length $|s_{n+1}| - 1$. If $s_{n+1} = \varepsilon$, then $s_1\#s_2\#\ldots\#s_n \in \mathcal{L}^{\triangleleft_1}$ by $s_n \in \mathcal{L}^\triangleleft$. Therefore, by the induction hypothesis, ($\heartsuit$) holds for s.

(2) $\Rightarrow$:

$$\begin{aligned} \mathcal{L}^{\triangleleft_1} &= (\mathcal{L}\#)^*\mathcal{L}^\triangleleft && (\mathcal{L}^{\triangleleft_1} \text{ def}) \\ &= (A^*\#)^*A^* && (\mathcal{L} = \mathcal{L}^\triangleleft = A^*) \\ &= (A \cup \#)^* \end{aligned}$$

(2) $\Leftarrow$: It is shown by the contraposition. It is easy to see that, if there is a string $s \notin \mathcal{L}$, then $s\# \notin \mathcal{L}^{\triangleleft_1}$ also holds. □

Lemma 4.1.16. *Let $\mathcal{L}$ be any language over A. Then the following hold.*

(1) $\mathcal{L}^{\triangleleft_1} = A^*_{\#} \iff \mu(\mathcal{L}^{\triangleleft_1}) = 1$.

(2) $\mathcal{L}^{\triangleleft_1} \neq A^*_{\#} \iff \mu(\mathcal{L}^{\triangleleft_1}) = 0$.

Proof. (1): By Proposition 4.1.12 (3).

(2)⇐: By $= \subseteq \simeq_{\mathrm{p}}$.

(2)⇒: Let s be a string not in $\mathcal{L}$ (such s always exists by Lemma 4.1.15(2)). Then one can see that for any two string, s' and s'', $s'\#s\#s'' \notin \mathcal{L}^{\triangleleft_1}$ holds by the construction of $\mathcal{L}^{\triangleleft_1}$. Therefore, by the infinite monkey theorem (i.e., for any string s, $\mu(A^*sA^*) = 1$ holds. see e.g., [80, Sec. 4.2.3]), $\mu(\mathcal{L}^{\triangleleft_1}) = 1 - \mu(A^* \setminus \mathcal{L}^{\triangleleft_1}) \leq 1 - \mu(A^*\#s\#A^*) = 0$. □

Corollary 4.1.17. *Let $\mathcal{L}$ be any language over A. Then the following hold.*

(1) $\mathcal{L} = A^* \iff \mathcal{L}^{\triangleleft_1} = A^*_{\#} \iff \mathcal{L}^{\triangleleft_1} \simeq_{\mathrm{p}} A^*_{\#}$.

(2) $\mathcal{L} \neq A^* \iff \mathcal{L}^{\triangleleft_1} \simeq_{\mathrm{p}} \emptyset$.

Proof. By Lemma 4.1.15(2) and Lemma 4.1.16. □

Corollary 4.1.17 gives a general approach to reduce from the universality problem to the p-universality (resp. p-emptiness) problem.

Expressing Prefix-closure Operator

In this subsubsection, we give several reductions from universality problems to p-universality (and p-emptiness) problems using Corollary 4.1.17. We give a translation from every REG t to a REG $t^{\triangleleft}$ such that $\mathcal{L}(t^{\triangleleft}) = \mathcal{L}(t)^{\triangleleft}$ holds.

Definition 4.1.18. The *prefix-closed REG* $t^{\triangleleft}$ of a REG t is a REG, inductively defined as follows:

(1) $a^{\triangleleft} := 1 \cup a$;

(2) $1^{\triangleleft} := 1$;

(3) $0^{\triangleleft} := 0$;

(4) $(t_1 \cdot t_2)^{\triangleleft} := [t_1^{\triangleleft}]_{\mathcal{L}(t_2)\neq\emptyset} \cup t_1 \cdot t_2^{\triangleleft}$;

(5) $(t_1 \cup t_2)^{\triangleleft} := t_1^{\triangleleft} \cup t_2^{\triangleleft}$;

(6) $(t^*)^\triangleleft := t^* \cdot t^\triangleleft$.

where $[t_1^\triangleleft]_{\mathcal{L}(t_2)\neq\emptyset}$ denotes the term $t_1^\triangleleft$ if $\mathcal{L}(t_2) \neq \emptyset$ and the term 0 otherwise.

Theorem 4.1.19. $\mathcal{L}(t^\triangleleft) = \mathcal{L}(t)^\triangleleft$.

Proof. It is proved by induction on the structure of t. The cases of (1), (2), (3), and (5) are easy.

(4): By the induction hypothesis, $\mathcal{L}((t_1 \cdot t_2)^\triangleleft) = [\mathcal{L}(t_1)^\triangleleft]_{\mathcal{L}(t_2)\neq\emptyset} \cup \mathcal{L}(t_1) \cdot \mathcal{L}(t_2)^\triangleleft$. If $s_1' \in \mathcal{L}(t_1)^\triangleleft$ and $\mathcal{L}(t_2) \neq \emptyset$, let $s_1 \in \mathcal{L}(t_1)$ be a string such that s_1' is a prefix of s_1 and let s_2 be a string in $\mathcal{L}(t_2)$. Then s_1' is a prefix of s_1s_2 and $s_1s_2 \in \mathcal{L}(t_1 \cdot t_2)$. Therefore $s_1' \in \mathcal{L}(t_1 \cdot t_2)^\triangleleft$. If $s_1 \in \mathcal{L}(t_1)$ and $s_2' \in \mathcal{L}(t_2)^\triangleleft$, let $s_2 \in \mathcal{L}(t_2)$ be a string such that s_2' is a prefix of s_2. Then s_1s_2' is a prefix of s_1s_2 and $s_1s_2 \in \mathcal{L}(t_1 \cdot t_2)$. Therefore $s_1s_2 \in \mathcal{L}(t_1 \cdot t_2)^\triangleleft$. Conversely, if $s' \in \mathcal{L}(t_1 \cdot t_2)^\triangleleft$, let $s \in \mathcal{L}(t_1 \cdot t_2)$ be a string such that s' is a prefix of s. Let $s_1 \in \mathcal{L}(t_1)$ and $s_2 \in \mathcal{L}(t_2)$ be strings such that $s = s_1s_2$. If $s' = s_1s_2'$, then $s' \in \mathcal{L}(t_1) \cdot \mathcal{L}(t_2)^\triangleleft$, and thus $s' \in \mathcal{L}((t_1 \cdot t_2)^\triangleleft)$. Otherwise, $s' \in \mathcal{L}(t_1)^\triangleleft$ and $\mathcal{L}(t_2) \neq \emptyset$ hold, and thus $s' \in \mathcal{L}((t_1 \cdot t_2)^\triangleleft)$.

(6): By the induction hypothesis, $L((t^*)^\triangleleft) = \mathcal{L}(t^*) \cdot \mathcal{L}(t)^\triangleleft$. If $s_1s_2' \in \mathcal{L}(t^*) \cdot \mathcal{L}(t)^\triangleleft$, let s_2 be a string such that s_2' is a prefix of s_2. Then s_1s_2' is a prefix of s_1s_2 and $s_1s_2 \in \mathcal{L}(t^*)$. Therefore $s_1s_2' \in \mathcal{L}(t^*)^\triangleleft$. Conversely, if $s' \in \mathcal{L}(t^*)^\triangleleft$, then there is a number n such that $s' \in \mathcal{L}(t^n)^\triangleleft$. Let $s_1 \in \mathcal{L}(t^{n-1})$ and $s_2' \in \mathcal{L}(t)^\triangleleft$ such that $s' = s_1s_2'$. Then $s_1 \in \mathcal{L}(t^*)$. Therefore $s_1s_2' \in \mathcal{L}(t^*) \cdot \mathcal{L}(t)^\triangleleft$, and thus $s' \in \mathcal{L}((t^*))^\triangleleft$. □

Corollary 4.1.20.

(1) There is a polynomial-time construction from every REG t to a REG $t^\triangleleft$ such that $\mathcal{L}(t^\triangleleft) = \mathcal{L}(t)^\triangleleft$ holds.

(2) There is a polynomial-time construction from every REG t to a REG $t^{\triangleleft_1}$ such that $\mathcal{L}(t^{\triangleleft_1}) = \mathcal{L}(t)^{\triangleleft_1}$ holds.

Corollary 4.1.21.

(1) The p-universality problem for REGs (resp. NFAs) is PSPACE*-hard.*

(2) The p-emptiness problem for REGs (resp. NFAs) is PSPACE*-hard.*

Proof. By Corollary 4.1.17 and that the universality problem for REGs is PSPACE-complete. □

We remark that this translation works for context-free grammars (CFGs) because there is a computable construction of context-free grammars for expressing the operations, concatenation ($\cdot$), Kleene-star ($\bullet^*$), and prefix closure ($\bullet^{\triangleleft}$).

Corollary 4.1.22.

*(1) The p-universality problem for CFGs is undecidable (*coRE*-hard).*

*(2) The p-emptiness problem for CFGs is undecidable (*RE*-hard).*

Proof. (1): By Corollary 4.1.17 (1) and that the universality problem for CFGs is coRE-hard [42][43, Lem. 3.1 (8)]. (2): By (1) and Corollary 4.1.17 (2). □

4.2 Upper bound: Descriptive Complexity for Automata

In this section we investigate about the computational complexity upper bounds of equational theories of Kleene algebra under some equivalences, where we assume each equivalence is defined by a logical formula. In particular we give the upper bound of the equational theory under p-equivalence using Lemma 4.0.3 and 4.1.13.

We now define the product of two NFAs. In this section we deal with equivalence relations on NFAs defined by logical formulas on such structures. In this section we assume that each NFA $\mathcal{A}$ has a total order $<$ on $|\mathcal{A}|$ because we rely on results of the descriptive complexity theory. (More formally, an NFA $\mathcal{A}$ over A is a structure over the signature $(<, \{\delta_a\}_{a\in A}, q^I, F)$, where $<^{\mathcal{A}}$ is a total order on $|\mathcal{A}|$; each $\delta_a^{\mathcal{A}}$ is a binary relation on $|\mathcal{A}|$; $(q^I)^{\mathcal{A}} \in |\mathcal{A}|$; and $F^{\mathcal{A}} \subseteq |\mathcal{A}|$.)

Definition 4.2.1 (Product)**.** The *product* of two NFAs $\mathcal{A}_1$ and $\mathcal{A}_2$, written $\mathcal{A}_1 \times \mathcal{A}_2$, is the structure $(|\mathcal{A}_1 \times \mathcal{A}_2|, <, \{(\delta_1)_a\}_{a\in A}, \{(\delta_2)_a\}_{a\in A}, q^I, F_1, F_2)$, where

(1) $|\mathcal{A}_1 \times \mathcal{A}_2| = Q_{\mathcal{A}_1} \times Q_{\mathcal{A}_2}$;

(2) $< = \{((q_1, q_2), (q_1', q_2')) \mid q_1 < q_1' \vee (q_1 = q_1' \wedge q_2 < q_2')\}$;

(3) $(\delta_1)_a = \{((q_1, q_2), (q_1', q_2)) \mid (q_1, q_1') \in (\delta_{\mathcal{A}_1})_a\}$;

(4) $(\delta_2)_a = \{((q_1, q_2), (q_1, q_2')) \mid (q_2, q_2') \in (\delta_{\mathcal{A}_2})_a\}$;

(5) $q^I = (q^I_{\mathcal{A}_1}, q^I_{\mathcal{A}_2})$;

(6) $F_1 = F_{\mathcal{A}_1} \times Q_{\mathcal{A}_2}$;

(7) $F_2 = Q_{\mathcal{A}_1} \times F_{\mathcal{A}_2}$.

Let us denote the formula $\bigvee_{a \in A} \exists q''.\delta_{1a}(q, q'') \wedge \delta_{2a}(q'', q')$ by $\mathrm{Reach1}(q, q')$, for short.

Definition 4.2.2 ($\mathcal{L}$-definable)**.** Let A be a nonempty finite alphabet and let $\mathcal{C}$ be a class of finite automata (e.g., DFA, unary NFA, ...). Let $\mathcal{L}$ be a logic (e.g., $\mathsf{FO}^<(\mathrm{TC})$, $\mathsf{SO}^<\forall$, ...). A relation $\simeq$ on $\wp(A^*)$ w.r.t. $\mathcal{C}$ is called *$\mathcal{L}$-definable* if there is a $\mathcal{L}$-sentence φ such that, for any two automata $\mathcal{A}_1$ and $\mathcal{A}_2$ in $\mathcal{C}$, $L(\mathcal{A}_1) \simeq L(\mathcal{A}_2)$ if and only if $\mathcal{A}_1 \times \mathcal{A}_2 \models \varphi$ holds.

Example 4.2.3. The normal equivalence $=$ w.r.t. DFA_A is $\mathsf{FO}(\mathrm{TC})$-definable by the following sentence:

$$(\forall q \mid \mathrm{Reach}(q^I, q)).F_1(q) \leftrightarrow F_2(q)$$

where $\mathrm{Reach}(q, q')$ denotes the formula $\mathrm{TC}_{q_0, q_0'}(\mathrm{Reach1}(q_0, q_0') \vee q_0 = q_0')(q, q')$.

Example 4.2.4. The normal equivalence $=$ w.r.t. NFA_A is $\mathsf{MSO}(\mathrm{TC})$-definable by the following sentence:

$$(\forall Q \mid \mathrm{Reach}(Q^I, Q)).F_1(Q) \leftrightarrow F_2(Q)$$

where $Q^I(q)$ denotes the formula $q = q^I$; $F_i(Q)$ denotes the formula $(\exists q \mid Q(q)).F_i(q)$ for $i = 1, 2$; $\mathrm{Reach}(Q, Q')$ denotes the formula $TC_{Q_0, Q_0'}(\mathrm{Reach1}(Q_0, Q_0') \vee \forall q.Q_0(q) \leftrightarrow Q_0'(q))(Q, Q')$; and $\mathrm{Reach1}(Q_0, Q_0')$ denotes the formula $\forall q'.Q_0'(q') \leftrightarrow ((\exists q \mid Q_0(q)).\mathrm{Reach1}(q, q'))$.

Note that the structure $\mathcal{A}_1 \times \mathcal{A}_2$ can be constructed from $\mathcal{A}_1$ and $\mathcal{A}_2$ in log-space. The following is immediate from results in descriptive complexity (see e.g., [44]).

Lemma 4.2.5 (e.g., [44])**.** *For any equational equivalence problem for an automata class $\mathcal{C}$ under a relation $\simeq$, the following hold:*

(1) If $\simeq$ w.r.t. $\mathcal{C}$ is $\mathsf{FO}^<(\mathrm{DTC})$-definable, then the equivalence problem (for $\mathcal{C}$ under $\simeq$) is in L*;*

(2) If $\simeq$ w.r.t. $\mathcal{C}$ is $\mathsf{FO}^<(\mathrm{TC})$-definable, then the equivalence problem is in NL*;*

(3) *If $\simeq$ w.r.t. $\mathcal{C}$ is $\mathsf{SO}^{<}\forall$-definable, then the equivalence problem is in* coNP*;*

(4) *If $\simeq$ w.r.t. $\mathcal{C}$ is $\mathsf{SO}^{<}(\mathrm{TC})$-definable (resp. $\mathsf{MSO}^{<}(\mathrm{TC})$-definable), then the equivalence problem is in* PSPACE *(resp.* NLINSPACE*).*

Proof Sketch. (1): By [44, Thm. 9.11]. (2): By [44, Cor. 9.22]. (3): By [44, Thm. 7.8] (called Fagin's theorem [31]). (4): By [44, Thm. 10.27 and Cor. 10.29]. □

Remark. In descriptive complexity, logics ordinary have the predicate BIT [44, Proviso 1.14]. However we omit the predicate by reason that the predicate is not needed in this thesis.

As a corollary to Lemma 4.2.5, the equivalence problem for DFA_A (resp. NFA_A) is decidable in NL (resp. NLINSPACE) by Example 4.2.3 and 4.2.4. In connection with Lemma 4.2.5, the following holds. Intuitively, the following lemma formalizes the technique to generate an algorithm for NFAs from an algorithm for DFAs based on the powerset construction.

Lemma 4.2.6. *Let A be a nonempty finite alphabet and let $\simeq$ be a relation on A^*. If $\simeq$ w.r.t. DFA_A is $\mathsf{FO}^{<}(\mathrm{TC})$-definable, then $\simeq$ w.r.t. NFA_A is $\mathsf{MSO}^{<}(\mathrm{TC})$-definable.*

Proof. We define a translation from an $\mathsf{FO}^{<}(\mathrm{TC})$-formula φ to an $\mathsf{MSO}^{<}(\mathrm{TC})$-formula φ^T as follows:

(i) $Q_{q^I}(q) :\equiv q = q^I$;

(ii) $(q_0 = q_0')^T :\equiv \forall q.Q_{q_0}(q) \leftrightarrow Q_{q_0'}(q)$;

(iii) $(q_0 < q_0')^T :\equiv (\exists q' \mid Q_{q_0'}(q')).(\forall q \mid \neg(q < q')).\neg Q_{q_0}(q)$;

(iv) $F_i(q_0)^T :\equiv (\exists q \mid Q_{q_0}(q)).F_i(q)$ for $i = 1, 2$;

(v) $(\delta_a)_i(q_0, q_0')^T :\equiv \forall q'.Q_{q_0'}(q') \leftrightarrow (\exists q \mid Q_{q_0}(q)).(\delta_a)_i(q, q')$ for $i = 1, 2$;

(vi) $(\forall q.\varphi)^T :\equiv \forall Q_q^1.\varphi^T$;

(vii) $(\exists q.\varphi)^T :\equiv \exists Q_q^1.\varphi^T$;

(viii) $(\varphi \wedge \psi)^T :\equiv \varphi^T \wedge \psi^T$;

(ix) $(\varphi \vee \psi)^T :\equiv \varphi^T \vee \psi^T$;

(x) $(\neg\varphi)^T :\equiv \neg\varphi^T$;

(xi) $\mathrm{TC}_{q,q'}(\varphi)(q_0,q_0')^T :\equiv \mathrm{TC}_{Q_q,Q_{q'}}(\varphi^T)(Q_{q_0},Q_{q_0'})$.

Then, $\mathcal{A}_1\times\mathcal{A}_2\models\varphi^T$ iff $\wp(\mathcal{A}_1)\times\wp(\mathcal{A}_2)\models\varphi$ holds, where the total order on $\wp(\mathcal{A}_1)\times\wp(\mathcal{A}_2)$ is defined according to the definition (iii).) It is easily proved by induction on the structure of φ. Therefore, $\mathcal{A}_1\times\mathcal{A}_2\models\varphi^T$ iff $\wp(\mathcal{A}_1)\times\wp(\mathcal{A}_2)\models\varphi$ iff $\mathcal{L}(\wp(\mathcal{A}_1))\simeq\mathcal{L}(\wp(\mathcal{A}_2))$ iff $\mathcal{L}(\mathcal{A}_1)\simeq\mathcal{L}(\mathcal{A}_2)$. Therefore $\simeq$ w.r.t. NFAs is defined by the $\mathsf{MSO}^<(\mathrm{TC})$-formula φ^T. □

Theorem 4.2.7. *The following hold.*

(1) The p-equivalence problem for unary DFAs is in L.

(2) The p-equivalence problem for DFAs is in NL.

(3) The p-equivalence problem for unary NFAs (resp. unary REGs) is in coNP.

(4) The p-equivalence problem for NFAs (resp. REGs) is in PSPACE *(more precisely, in* NLINSPACE*).*

Proof. (1)(3): Let us recall the formula in Lemma 4.1.9. For (1), p-equivalence w.r.t. unary DFAs is defined by the following $\mathsf{FO}(\mathrm{DTC})$-formula:

$$\neg((\exists q\mid F_1(q)\nleftrightarrow F_2(q)).(\mathrm{Reach}^+(q^I,q)\vee q^I=q)\wedge\mathrm{Reach}^+(q,q))$$

where $\mathrm{Reach}^+(q_0,q_0')$ denotes the formula $\mathrm{DTC}_{q,q'}(\mathrm{Reach1}(q,q'))(q_0,q_0')$.

For (3), p-equivalence w.r.t. unary NFAs is defined by the following $\mathsf{SO}^<\forall$-formula[1]:

$$\neg((\exists Q^1\mid(\exists q.Q(q)\wedge F_1(q))\nleftrightarrow(\exists q.Q(q)\wedge F_2(q))).(\mathrm{Reach}^+(Q^I,Q)\vee Q^I=Q)\wedge\mathrm{Reach}^+(Q,Q))$$

where $Q^I(q)$ denotes the formula $q^I=q$ and $\mathrm{Reach}^+(Q_0,Q_0')$ denotes the following formula:

$$\exists X^3.((\forall i\mid i<\overline{\max}).X_{i+1}=X_i\cdot X_i\vee X_{i+1}=X_i\cdot X_i\cdot X_0)\wedge((\exists i\mid i\leq\overline{\max}).Q_0'=X_i(Q_0)$$

where (i) the symbol $\cdot$ denotes the composition of binary relations (actually we can easily replace the above formula by an FO formula without the symbol $\cdot$, see

[1] This formula is not an $\mathsf{SO}^<\forall$-formula, but its prenex normal form is.

Theorem 2.3.5); (ii) $X_i(q,q')$ denotes $X(i,q,q')$; (iii) $X_0(q,q')$ denotes $\mathrm{Reach1}(q,q')$; (iv) $Q'_0 = X_i(Q_0)$ denotes the formula $\forall q'.Q'_0(q') \leftrightarrow (\exists q \mid Q_0(q)).X_i(q,q'))$.

$\mathrm{Reach}^+(Q_0,Q'_0)$ is defined to satisfy that $\mathrm{Reach}^+(Q_0,Q'_0)$ holds if and only if $(\exists k \mid 1 \leq k).Q'_0 = X_0^k(Q_0)$. In the right hand of the above formula, without loss of generality, we can assume that $k \leq 2^{\overline{\max}}$ because the pattern of Q'_0 (i.e., the number of monadic predicates) is at most $2^{\overline{\max}}$. In fact every X_0^k $(1 \leq k \leq 2^{\overline{\max}})$ can be expressed by a binary relation X_i of a sequence $(X_0,\ldots,X_{\overline{\max}})$. Every X_i denotes one of the binary relations $X_0^{2^i}, X_0^{2^i+1},\ldots,X_0^{2^{i+1}-1}$. Therefore $\mathrm{Reach}^+(Q_0,Q'_0)$ holds if and only if $(\exists k \mid 1 \leq k \leq 2^{\overline{\max}}).Q'_0 = X_0^k(Q_0)$ if and only if $(\exists k \mid 1 \leq k).Q'_0 = X_0^k(Q_0)$. By Lemma 4.2.6 and Lemma 4.2.5 (1)(3), these parts are proved.

(2)(4): Let us recall the formula in Lemma 4.0.3. By using this, p-equivalence w.r.t. DFAs is defined by a $\mathsf{FO}(\mathrm{TC})$-formula as follows:

$$\neg((\exists q \mid F_1(q) \nleftrightarrow F_2(q)).\mathrm{Reach}(q^I,q) \wedge (\forall q' \mid \mathrm{Reach}(q,q')).\mathrm{Reach}(q',q))$$

where $\mathrm{Reach}(q,q') \equiv \mathrm{TC}_{q,q'}(\mathrm{Reach1}(q,q') \vee q = q')$. By Lemma 4.2.6 and Lemma 4.2.5(2)(4), these parts are proved.

For the cases of REGs, it is shown by translating from REGs to NFAs (see Figure 2.6). (Be slightly careful to show that the problem for REGs is in NLINSPACE because the length of the encoded string of the (fully) translated NFA is not linear of the size of a regular expression.) □

4.3 Lower Bound: The p-universality Problem

In this section, we prove that every weak equivalence subsuming p-equivalence is computationally hard. The key idea is to construct a reduction to ones recognizing (regular) languages $\mathcal{L}_1$ and $\mathcal{L}_2$ satisfying the following: $\mathcal{L}_1 = \mathcal{L}_2 \iff \mathcal{L}_1 \simeq_{\mathrm{p}} \mathcal{L}_2$. Such reduction is applicable to any binary relation between the normal equivalence and the p-equivalence. In particular, we prove the complexity results in Theorem 4.2.7 are tight. More precisely, we prove the hardness for the p-universality problem, that is the problem to decide whether a given language L is equivalent to the

language A^* under p-equivalence. (As one can see, this problem is a subproblem of the p-equivalence problem.)

Theorem 4.3.1.

(1) The p-universality problem for unary DFAs is L*-complete.*

(2) The p-universality problem for DFAs is NL*-complete.*

(3) The p-universality problem for unary REGs (resp. unary NFAs) is coNP*-complete.*

(4) The p-universality problem for REGs (resp. NFAs) is PSPACE*-complete.*

Proof. The upper bounds have already been shown in Theorem 4.2.7. We now prove the lower bounds.

(1)(2): A *finite unlabelled digraph* (graph, for short) G is a structure $G = (|G|, E, s, t)$, where $E \subseteq |G|^2$ and $s, t \in |G|$. G is called *deterministic* if $\#\{v' \mid (v, v') \in E\} = 1$ holds for any $v \in |G|$. In this proof we assume that (a) $s \neq t$; (b) for any $v \in |G|$, $(v, v) \notin E$. (These restrictions do not loss the hardness of the reachability problem.)

(1): We reduce from the *deterministic graph reachability problem* [44, Sec 3.3.]. Let $REACH_d$ be the set of deterministic graphs such that $(s, t) \in E^*$ holds.

We construct a first-order reduction (see [44, Ch 3.]) from a deterministic graph G to an unary DFA $\mathcal{A}_G$ such that $G \in REACH_d \iff L(\mathcal{A}_G) \not\simeq_{\mathrm{p}} \emptyset$. Let $\mathcal{A}_G$ be the unary DFA defied as $\mathcal{A}_G = (|\mathcal{A}_G|, \delta, q^I, F)$, where (i) $|\mathcal{A}_G| = |G|$; (ii) $\delta(v) = v'$ if $v \neq t$, where v' is the unique node such that $(v, v') \in E$; (iii) $\delta(t) = s$; (iv) $q^I = s$; (v) $F = \{t\}$. $\mathcal{A}_G$ and the complemented DFA $\mathcal{A}_G^{\mathsf{C}} = (|\mathcal{A}_G|, \delta, q^I, |\mathcal{A}_G| \setminus F)$ are first-order reducible from G. Then $G \in REACH_d \iff \mathcal{L}(\mathcal{A}_G) \neq \emptyset$ and $\mathcal{L}(\mathcal{A}_G)$ is empty or infinite. Then,

$$
\begin{aligned}
G \in REACH_d &\iff \mathcal{L}(\mathcal{A}_G) \neq \emptyset && \text{(By the construction of } \mathcal{A}_G\text{)} \\
&\iff \mathcal{L}(\mathcal{A}_G) \not\simeq_{\mathrm{f}} \emptyset && (\mathcal{L}(\mathcal{A}_G) \text{ is empty or infinite}) \\
&\iff \mathcal{L}(\mathcal{A}_G) \not\simeq_{\mathrm{p}} \emptyset && \text{(Propopsition 4.1.5)} \\
&\iff \mathcal{L}(\mathcal{A}_G^{\mathsf{C}}) \not\simeq_{\mathrm{p}} A^* && (\mathcal{A}_G^{\mathsf{C}} \text{ is the complemented DFA of } \mathcal{A}_G)
\end{aligned}
$$

By that $REACH_d$ is L-complete and that the class L is obviously closed under complement, this part is proved.

(2): We reduce from the *graph reachability problem* [44, Sec. 3.3] (the proof of this part is based on [46, Thm. 26]). Let $REACH$ be the set of graphs G such that $(s,t) \in E^*$ holds. We construct a DFA $\mathcal{A}_G$ from a graph G such that $G \in REACH \iff \mathcal{L}(\mathcal{A}_G) \not\simeq_{\mathrm{p}} \emptyset$ holds. Let $\mathcal{A}_G$ be the DFA over $|G|$ defied as $\mathcal{A}_G = (|\mathcal{A}_G|, \{\delta_v\}_{v\in|G|}, q^I, F)$, where (i) $|\mathcal{A}_G| = |G| \cup \{\mathrm{Err}\}$; (ii) $\delta_{v'}(v) = v'$ if $v \neq t$ and $(v,v') \in E$; $\delta_{v'}(t) = t$; $\delta_{v'}(v) = \mathrm{Err}$ otherwise; (iii) $q^I = s$; (iv) $F = \{t\}$. Then $G \in REACH$ if and only if $L(\mathcal{A}_G) \neq \emptyset$, and $L(\mathcal{A}_G^{\mathsf{C}})$ is prefix-closed. Then,

$$
\begin{aligned}
G \in REACH &\iff \mathcal{L}(\mathcal{A}_G) \neq \emptyset && \text{(By the construction of } \mathcal{A}_G\text{)} \\
&\iff \mathcal{L}(\mathcal{A}_G^{\mathsf{C}}) \neq A^* && (\mathcal{A}_G^{\mathsf{C}} \text{ is the complemented DFA of } \mathcal{A}_G) \\
&\iff \mathcal{L}(\mathcal{A}_G^{\mathsf{C}}) \not\simeq_{\mathrm{p}} \emptyset && (\mathcal{L}(\mathcal{A}_G^{\mathsf{C}}) \text{ is prefix-closed and Prop.4.1.12(3)})
\end{aligned}
$$

where $\mathcal{A}_G^{\mathsf{C}}$ is the complemented DFA of $\mathcal{A}_G$. By that $REACH$ is NL-complete and that $\mathcal{A}_G^{\mathsf{C}}$ is log-space reducible from G, this part is proved. Note that the class NL is closed under complement by Immerman-Szelepcsényi Theorem [45, 82].

(3): We reduce from the *3-SAT problem* [49, p.95 11]. This part is proved by using the same reduction as [81, Thm. 6.1]. Let $p_1, p_2, \ldots, p_n$ be the sequence of prime numbers (i.e., $2, 3, 5, \ldots$), and let $A = \{0\}$. Let E_φ be the regular expression of a 3-CNF formula φ defined as $E_\varphi = E_0 \cup \bigcup_{k=1}^{m} E_k$, where $E_0, \ldots, E_m$ are defined in [81, Thm. 6.1]. Then the following holds by the construction of E_φ ($\heartsuit$): $A^* \setminus \mathcal{L}(E_\varphi) \neq \emptyset \iff \#(A^* \setminus \mathcal{L}(E_\varphi))$ is infinite. It is because, $0^{i_1} \in \mathcal{L}(E_\varphi) \iff 0^{i_2} \in \mathcal{L}(E_\varphi)$ holds for any two integers, i_1 and i_2, such that $i_1 \equiv i_2 (\mathrm{mod} \prod_{\mathrm{k}=1}^{\mathrm{n}} \mathrm{p_k})$, where n is defined in [81, Thm. 6.1]. Then,

$$
\begin{aligned}
\varphi \text{ is not satisfiable} &\iff \mathcal{L}(E_\varphi) = A^* && ([81, \text{Thm. } 6.1]) \\
&\iff \mathcal{L}(E_\varphi) \simeq_{\mathrm{f}} A^* && (\heartsuit) \\
&\iff \mathcal{L}(E_\varphi) \simeq_{\mathrm{p}} A^* && (\text{Proposition } 4.1.5)
\end{aligned}
$$

By Theorem 4.2.7(3) and that the 3-SAT problem is NP-complete, this part is proved.

(4): It has already been shown in Corollary 4.1.21. Appendix B gives another reduction, that is a log-lin space reduction from the membership problem for nondeterministic linear-space bounded Turing machines. This reduction shows that the p-universality problem for REGs is also NLINSPACE-hard (under log-lin space reductions). □

More strongly, the proof of Theorem 4.3.1 is applicative to every weak equivalence subsuming p-equivalence.

Corollary 4.3.2. *Let $\simeq$ be a binary relation such that $= \subseteq \simeq \subseteq \simeq_{\mathrm{p}}$.*

(1) The universality problem for unary DFAs under $\simeq$ is L*-hard.*

(2) The universality problem for DFAs under $\simeq$ is NL*-hard.*

(3) The universality problem for unary NFAs (resp. unary REGs) under $\simeq$ is coNP*-hard.*

(4) The universality problem for NFAs (resp. REGs) under $\simeq$ is PSPACE*-hard.*

Proof. We only prove for (4). Let us recall the REG $t^{\lhd_1}$ in Corollary 4.1.21. Note that $\mathcal{L}(t^{\lhd_1}) \simeq_{\mathrm{p}} A_{\#}^* \Longrightarrow \mathcal{L}(t^{\lhd_1}) = A_{\#}^*$ by Proposition 4.1.12. Then, by $= \subseteq \simeq \subseteq \simeq_{\mathrm{p}}$, $\mathcal{L}(t^{\lhd_1}) = A_{\#}^* \Longrightarrow \mathcal{L}(t^{\lhd_1}) \simeq A_{\#}^* \Longrightarrow \mathcal{L}(t^{\lhd_1}) \simeq_{\mathrm{p}} A_{\#}^* \Longrightarrow \mathcal{L}(t^{\lhd_1}) = A_{\#}^*$. Therefore, the reduction is also applicable to the universality problem under $\simeq$. (1), (2), and (3) are also proved in the same way as (4). □

For example, $\simeq_{\mathrm{f}}$ and $\simeq_{\mathcal{L}}$ satisfies the condition in Corollary 4.3.2 if $\mathcal{L}$ satisfies $\mu(\mathcal{L}) = 0$. From this, the hardness of the universality problem under $\simeq_{\mathrm{f}}$ (resp. $\simeq_{\mathcal{L}}$) is also given by Corollary 4.3.2.

As a corollary to Theorem 4.2.7 and 4.3.1, the p-equivalence problem is also hard.

Corollary 4.3.3.

(1) The p-equivalence problem for unary DFAs is L*-complete.*

(2) The p-equivalence problem for DFAs is NL*-complete.*

(3) The p-equivalence problem for unary REGs (resp. unary NFAs) is coNP*-complete.*

(4) The p-equivalence problem for REGs (resp. NFAs) is PSPACE*-complete.*

The proof of Theorem 4.3.1 is also applicable to show the hardness of testing the zero-one law for finite word structures (see e.g., [80, p. 9]). Let us denote the problem by *the zero-one problem* (i.e., the zero-one problem to decide whether a given language $\mathcal{L}$ satisfies that $\mu(\mathcal{L}) = 0 \vee \mu(\mathcal{L}) = 1$, or not).

As related work, the zero-one problem for DFAs is solvable in linear time [80, Thm. 5.1.1], where the alphabet A is fixed. We can show the computational complexity of this problem as a corollary.

Corollary 4.3.4.

(1) The zero-one problem for unary DFAs is L*-complete.*

(2) The zero-one problem for DFAs is NL*-complete.*

(3) The zero-one problem for unary NFAs (resp. unary REGs) are coNP*-complete.*

(4) The zero-one problem for NFAs (resp. REGs) is PSPACE*-complete.*

Proof. Note that the zero-one problem can be solved by the two p-equivalence problems as $\mathcal{L} \simeq_{\mathrm{p}} \emptyset \vee \mathcal{L} \simeq_{\mathrm{p}} A^*$. Therefore the upper bounds are derived from Theorem 4.2.7. For lower bound, we suffice to show that, if every language L in Theorem 4.3.1 satisfies the zero-one law (i.e., $\mu(\mathcal{L}) = 0$ or $\mu(\mathcal{L}) = 1$), then $\mu(\mathcal{L}) \neq 0$.

(1): Recall the reduction in Theorem 4.3.1(1). Then $\mu(\mathcal{L}(\mathcal{A}_G^{\mathsf{C}})) \neq 0$ because, there are infinitely many i such that the position in i th step is not equal to t (note that $s \neq t$).

(2): Recall the reduction in Theorem 4.3.1(2). Then $\mathcal{L}(sA^*) \subseteq \mathcal{L}(\mathcal{A}_G^{\mathsf{C}})$ holds by that $(s, s) \notin E$ and $s \neq t$. Therefore $\mu(\mathcal{L}(\mathcal{A}_G^{\mathsf{C}})) \neq 0$.

(3): Recall the reduction in Theorem 4.3.1(3). Then $\mu(\mathcal{L}(E)) \neq 0$ holds by that $\#\mathcal{L}(E)$ is always infinite.

(4): Recall the reduction in Theorem 4.3.1(4). Then $\mu(\mathcal{L}(t_M^s)) \neq 0$ holds by that $\mathcal{L}(\#\#A^*) \subseteq \mathcal{L}(t_M^s)$. □

4.3.1 The p-emptiness Problem

In this subsection we show that the p-emptiness problem is also as hard as the p-universality problem, whereas the emptiness problem is easier than the universality

problem under the normal equivalence. The following holds for the emptiness problem under the normal equivalence.

Proposition 4.3.5.

*(1) The emptiness problem for REGs (resp. unary REGs) is (*DLOGTIME*-uniform)* NC^1*-complete.*

(2) The emptiness problem for unary DFAs is L*-complete.*

(3) The emptiness problem for DFAs (resp. NFAs, unary NFAs) is NL*-complete.*

Proof. (1): This problem is solved by the following function Ex from $\mathcal{T}_A^{KA}$ to $\{true, false\}$:

(i) $\mathrm{Ex}(a) := true$;

(ii) $\mathrm{Ex}(1) := true$;

(iii) $\mathrm{Ex}(0) := false$;

(iv) $\mathrm{Ex}(t_1 \cup t_2) := \mathrm{Ex}(t_1) \vee \mathrm{Ex}(t_2)$;

(v) $\mathrm{Ex}(t_1 \cdot t_2) := \mathrm{Ex}(t_1) \wedge \mathrm{Ex}(t_2)$;

(vi) $\mathrm{Ex}(t_1^*) := true$.

It is easy to see that $\mathrm{Ex}(t) \iff \mathcal{L}(t) \neq \emptyset$ holds by induction on the structure of t.

Recall that the membership problem for every parenthesis context-free language is in L [59, Thm. 1]. More precisely, this problem is in ALOGTIME [22, Thm. 8] (ALOGTIME is equivalent to DLOGTIME-uniform NC^1 [65, Lem. 6.2]). Also recall that the *BSVP* (boolean sentence value problem) is a NC^1-complete problem [22, Thm. 9] [23, Thm. 5]. The BSVP is the problem to decide whether the value of a given boolean sentence (i.e., a formula formed from 1 (for *true*), 0 (for *false*), $\wedge$, $\vee$, and $\neg$) is *true* or *false*. The BSVP is also NC^1-complete even if the problem is restricted to connectives $\wedge$ and $\vee$ (see e.g., [21]).

The above function Ex can be expressed by a parenthesis context-free grammar like as [22, Sec.6], and thus this problem is in NC^1. Conversely, there is a first-order reduction (i.e., DLOGTIME-uniform AC^0 reduction [65, Thm. 8.1]) from every BSVP

to an emptiness problem for REGs. Let consider the following translation from every boolean sentence φ to an regular expression t_φ inductively defined as follows:

(1) $t_0 := 0$;

(2) $t_1 := 1$;

(3) $t_{\varphi_1 \vee \varphi_2} := t_{\varphi_1} \cup t_{\varphi_2}$;

(4) $t_{\varphi_1 \wedge \varphi_2} := t_{\varphi_1} \cdot t_{\varphi_2}$.

Then φ is *true* if and only if $\mathcal{L}(t_\varphi) \neq \emptyset$. Therefore this problem is NC^1-complete.

(2)(3): Each problems is essentially equivalent to the (deterministic or nondeterministic) graph reachability problems by forgetting labels of transitions. □

Now we consider about the p-emptiness problem.

Theorem 4.3.6 (cf. Proposition 4.3.5)**.**

(1) The p-emptiness problem for unary REGs is NC^1*-complete.*

(2) The p-emptiness problem for unary DFAs is L*-complete.*

(3) The p-emptiness problem for unary NFAs is NL*-complete.*

(4) The p-emptiness problem for DFAs is NL*-complete.*

(5) The p-emptiness problem for REGs (resp. NFAs) is PSPACE*-complete.*

Proof. (2)(4): These have already been proved in Theorem 4.3.1(1)(2).

(3): The NL-hardness is shown by using the graph reachability problem. Moreover this problem is in NL because the formula in Lemma 4.1.9 is easily written in $\mathsf{FO(TC)}$.

(5): By Theorem 4.2.7(4) and Corollary 4.1.21(2).

(1): Let us recall that $\mathcal{L} \not\approx_{\mathrm{p}} \emptyset$ if and only if $\#\mathcal{L}$ is infinite by Proposition 4.1.5. We define the three functions from $\mathcal{T}_A^{KA}$ to $\{true, false\}$, denoted by $\mathrm{Ex}, \mathrm{Ex}^+, \mathrm{Ex}^\infty$, satisfying the following:

$$\begin{aligned} \mathrm{Ex}(t) &\iff \mathcal{L}(t) \neq \emptyset; \\ \mathrm{Ex}^{+}(t) &\iff \mathcal{L}(t) \setminus \{\varepsilon\} \neq \emptyset; \\ \mathrm{Ex}^{\infty}(t) &\iff \#(\mathcal{L}(t)) \text{ is infinite.} \end{aligned}$$

The function Ex has already been defined in Proposition 4.3.5(1). The function Ex^{+} and Ex^{∞} can be defined as follows:

(i) $\mathrm{Ex}^{+}(a) = \mathit{true}$;

(ii) $\mathrm{Ex}^{+}(1) = \mathit{false}$;

(iii) $\mathrm{Ex}^{+}(0) = \mathit{false}$;

(iv) $\mathrm{Ex}^{+}(t_1 \cup t_2) = \mathrm{Ex}^{+}(t_1) \vee \mathrm{Ex}^{+}(t_2)$;

(v) $\mathrm{Ex}^{+}(t_1 \cdot t_2) = (\mathrm{Ex}^{+}(t_1) \wedge \mathrm{Ex}(t_2)) \vee (\mathrm{Ex}(t_1) \wedge \mathrm{Ex}^{+}(t_2))$;

(vi) $\mathrm{Ex}^{+}(t_1^*) = \mathrm{Ex}^{+}(t_1)$;

(i) $\mathrm{Ex}^{\infty}(a) = \mathit{false}$;

(ii) $\mathrm{Ex}^{\infty}(1) = \mathit{false}$;

(iii) $\mathrm{Ex}^{\infty}(0) = \mathit{false}$;

(iv) $\mathrm{Ex}^{\infty}(t_1 \cup t_2) = \mathrm{Ex}^{\infty}(t_1) \vee \mathrm{Ex}^{\infty}(t_2)$;

(v) $\mathrm{Ex}^{\infty}(t_1 \cdot t_2) = (\mathrm{Ex}^{\infty}(t_1) \wedge \mathrm{Ex}(t_2)) \vee (\mathrm{Ex}(t_1) \wedge \mathrm{Ex}^{\infty}(t_2))$;

(vi) $\mathrm{Ex}^{\infty}(t_1^*) = \mathrm{Ex}^{+}(t_1)$.

Let $\mathrm{F}(t) = (\mathrm{Ex}(t), \mathrm{Ex}^{+}(t), \mathrm{Ex}^{\infty}(t))$. Then the function F can be expressed by a parenthesis context-free grammar like as [22, Sec.6], and thus the problem is in NC^1. Moreover there is a first-order reduction from the BSVP. Let us consider the translation from a given boolean sentence φ to a regular expression t_φ inductively defined as follows:

(1) $t_0 := 0^*$;

(2) $t_1 := a^*$;

(3) $t_{\varphi_1 \vee \varphi_2} := (t_{\varphi_1} \cup t_{\varphi_2})^*$;

(4) $t_{\varphi_1 \wedge \varphi_2} := (t_{\varphi_1} \cdot t_{\varphi_2})^*$.

It is easy to see that φ is *true* if and only if $\#\mathcal{L}(t_\varphi)$ is infinite (i.e., $\mathcal{L}(t_\varphi) \not\simeq_{\mathrm{p}} \emptyset$) by induction on the structure of φ. Actually this translation induces a first-order reduction. Therefore the p-emptiness problem for unary REGs is NC^1-complete. □

Table 4.1 is the summary of the computational complexity results.

	$\#A = 1$			$\#A \geq 2$		
	REG	DFA	NFA	REG	DFA	NFA
$=$ [81][46, Thm. 26]	coNP	L	coNP	PSPACE	NL	PSPACE
$\simeq_{\mathrm{p}}$ (Cor. 4.3.3)	coNP	L	coNP	PSPACE	NL	PSPACE
universality [81][46, Thm. 26]	coNP	L	coNP	PSPACE	NL	PSPACE
p-universality (Thm. 4.3.1)	coNP	L	coNP	PSPACE	NL	PSPACE
emptiness [81][46, Thm. 26]	NC^1	L	NL	NC^1	NL	NL
p-emptiness (Thm. 4.3.6)	NC^1	L	NL	PSPACE	NL	PSPACE

TABLE 4.1: The results for p-equivalence in language semantics

Remark. If A is finite, every language problem can be reduced to the case of $\#A = 2$ by Proposition 4.1.10. Be slightly careful in Theorem 4.3.1 (3) because the size of the alphabet of reduced DFAs is not bounded, but the reduction works all right in log-space by that the size of the alphabet of every reduced DFA is linear of the size of the DFA.

4.4 p-equivalence in Relational Semantics

In this section we investigate the p-equivalence of Kleene algebra in relational semantics. $\simeq^{\mathrm{L}}_{\mathrm{p}}$ denotes the p-equivalence $\simeq_{\mathrm{p}}$ defined in Definition 4.1.1 for the sake of clarity. We define μ^{R}_n as follows:

$$\mu^{\mathrm{R}}_n(t_1 = t_2) = \frac{\#\{M \in REL^A_n \mid M \models t_1 = t_2\}}{\#REL^A_n}.$$

$\mu^R(t_1 = t_2)$ denotes $\lim_{n\to\infty} \mu^{\mathrm{R}}_n(t_1 = t_2)$. This definition is just the asymptotic probability used for the zero-one law in finite model theory (see e.g., [55, Sec.12]). Then

the p-equivalence in relational semantics, written $\simeq_{\mathrm{p}}^{\mathrm{R}}$, is defined as $t_1 \simeq_{\mathrm{p}}^{\mathrm{R}} t_2 \iff \mu(t_1 = t_2) = 1$. The following are shown in the same manner as Proposition 4.1.4.

Proposition 4.4.1. *$\simeq_{\mathrm{p}}^{\mathrm{R}}$ is an equivalence relation.*

The two p-equivalences in language semantics and relational semantics are different, whereas the two normal equivalences in these semantics are equivalent (Theorem 2.3.14).

Proposition 4.4.2. *$\simeq_{\mathrm{p}}^{\mathrm{L}}$ and $\simeq_{\mathrm{p}}^{\mathrm{R}}$ are incomparable, i.e., both $\simeq_{\mathrm{p}}^{\mathrm{L}} \subseteq \simeq_{\mathrm{p}}^{\mathrm{R}}$ and $\simeq_{\mathrm{p}}^{\mathrm{R}} \subseteq \simeq_{\mathrm{p}}^{\mathrm{L}}$ fail.*

Proof. $\simeq_{\mathrm{p}}^{\mathrm{L}} \not\subseteq \simeq_{\mathrm{p}}^{\mathrm{R}}$: The equational formula $\varphi \equiv A = 0$ is a counterexample. It is easy to see that $A \simeq_{\mathrm{p}}^{\mathrm{L}} 0$ holds. However $A \simeq_{\mathrm{p}}^{\mathrm{R}} 0$ does not hold because $\mu_n^{\mathrm{R}}(\varphi) = \frac{\#\{M \in REL_n^A \mid (R_M)_a = \emptyset \text{ for any } a \in A\}}{\#REL_n^A} = \frac{1}{\#REL_n^A}$.

$\simeq_{\mathrm{p}}^{\mathrm{R}} \not\subseteq \simeq_{\mathrm{p}}^{\mathrm{L}}$: The equational formula $\varphi \equiv (AA)^* = A^*$ is a counterexample. It is easy to see that $(AA)^* \simeq_{\mathrm{p}}^{\mathrm{L}} A^*$ does not hold. However $(AA)^* \simeq_{\mathrm{p}}^{\mathrm{R}} A^*$ holds. For the sake of brevity, we assume that $\#A = 1$ and $A = \{a\}$. We prove that the following holds for almost any digraph: for any two nodes, s and t, if t is reachable from s, then t is also reachable form s by an *even length path*. Let ψ be the formula $(\forall v.\forall v'.((R_M)_a)^*(v, v')) \wedge (\exists v''.(R_M)_a(v'', v''))$. Intuitively, ψ denotes that the structure M is strongly connected and there is a self-loop in M. Then we show the following: (i) $M \models \psi \Longrightarrow M \models \varphi$, and thus $\mu_n^{\mathrm{R}}(\varphi) \geq \mu_n^{\mathrm{R}}(\psi)$; (ii) $\mu^{\mathrm{R}}(\psi) = 1$. (i): Let v'' be a node having self-loop. Then there is always an even length path from any node v to any node v' by considering a path via the node v''. (ii): This is an analogy of that almost all digraphs are strongly connected (more strongly, almost all digraphs have diameter 2, see e.g., [14, Cor. 7.1]). Therefore $\mu^{\mathrm{R}}(\varphi) = 1$. □

In relational semantics, every equational formula of Kleene algebra satisfies the zero-one law and the equivalence problem under $\simeq_{\mathrm{p}}^{\mathrm{R}}$ is decidable.

Proposition 4.4.3. *In relational semantics, the following hold.*

(1) *Every equational formula φ of Kleene algebra satisfies the zero-one law (i.e, $\mu^{\mathrm{R}}(\varphi) = 0$ or $\mu^{\mathrm{R}}(\varphi) = 1$ holds).*

(2) *The equivalence problem for Kleene algebra terms under $\simeq_{\mathrm{p}}^{\mathrm{R}}$ is decidable in* PSPACE.

Proof. Let us recall that every equational formula of Kleene algebra can be expressed by an $\mathsf{FO3(TC)}$ sentence (Corollary 2.3.11). Therefore these are immediate from [10, Cor. 3.2 and Thm. 6.1]. □

The following question remains open: in relational semantics, what is the complexity of the equivalence problem for Kleene algebra under $\simeq_{\mathrm{p}}^{\mathrm{R}}$? (Is the problem much easier than the normal equivalence problem?)

4.5 Conclusion and Future Work

In this chapter we introduced p-equivalence and gave the computational complexity of the equational theory for several models of regular languages under p-equivalence. Moreover we showed that the p-universality problem (resp. the p-emptiness problem) for context-free grammars is undecidable. As shown in Table 4.1, in the class of regular languages, the equational theory under p-equivalence is not easier than one under the normal equivalence. One of the possible future works is to study the p-equivalence problem in relational semantics. Another direction is to characterize hyper-minimization under p-equivalence like [5, Thm. 3.4].

Chapter 5

Concluding Remarks

In this thesis, we studied decidability and computational complexity of several extensions of Kleene algebra.

In chapter 3, we showed that the equational theory of *Kleene allegories* (Kleene algebra in relational semantics with relational intersection and relational converse) is decidable and EXPSPACE-complete. In connection with this, several extensions are considerable as future work. One extension is to add the universal relation symbol $\top$ in [17]. We remarked that Kleene allegories with $\top$ is equivalent to existential positive $\mathsf{FO3(TC)}$ in the sense of expressive power of binary relations (Corollary 2.3.11). Concerning this, it is interesting to study the query equivalence problem of existential positive $\mathsf{FO(TC)}$. Another extension is to add restricted negations preserving decidability, e.g., boolean tests in Kleene algebra with tests [53], atomic negation (see e.g., [57]), antidomain [28], unary negation [78], or guarded negation [6]. It may be also interesting to extend with other operators used in logic and (finite) model theory like least fixed point operator (LFP) or second-order quantifier.

In chapter 4, we studied the almost surely equivalences (called *p-equivalences*) on Kleene algebra terms over word structures and over structures of binary relations. We showed that, for Kleene algebra terms over word structures, the almost surely equivalence problem is still PSPACE-hard (more precisely, NLINSPACE-hard) as same as the equivalence problem. Moreover we gave a reduction from every almost surely equivalence problem to an equivalence problem (Corollary 4.1.17). This reduction implies that the almost surely equivalence (resp. universality, emptiness) problem

is not easier than the universality problem in some language classes (e.g., the class of regular languages, the class of context-free languages) over word structures. This negative fact is in contrast to the following fact for first-order logic: the almost surely validity problem is decidable, whereas the finite validity problem is undecidable. In this chapter, we also investigated the almost surely equivalence on Kleene algebra terms over structures of binary relations. Over structures of binary relations, every equational formula of Kleene algebra terms satisfies the zero-one law, whereas it is not the case over word structures. The almost surely equivalence problem for Kleene algebra terms over structures of binary relations is decidable, but the computational complexity remains open.

Appendix A

Proof of Theorem 2.3.7

We prove the following by induction on the pair $(\|\varphi\|, -\#(\{s,t\}))$: $[\![\varphi]\!]^{s,t}_{\mathcal{A}} = T_{s,t}(\varphi)^{\mathcal{A}}$.

(1) $s = t\ (= u)$:

$$
\begin{aligned}
&(m_u, m'_u) \in [\![\varphi]\!]^{u,u}_{\mathcal{A}} \\
&\iff (\exists m_v.(m_u, m_v) \in [\![\varphi]\!]^{u,v}_{\mathcal{A}}) \wedge m_u = m'_u \\
&\iff (\exists m_v.(m_u, m_v) \in [\![\varphi]\!]^{u,v}_{\mathcal{A}} \wedge (m_v, m'_u) \in |\mathcal{A}|^2) \wedge (m_u, m'_u) \in 1^{\mathcal{A}} \\
&\iff (m_u, m'_u) \in ([\![\varphi]\!]^{u,v}_{\mathcal{A}} \cdot \top^{\mathcal{A}}) \cap 1^{\mathcal{A}} \\
&\iff (m_u, m'_u) \in ((T_{u,v}(\varphi) \cdot \top) \cap 1)^{\mathcal{A}} && \text{(I.H.)} \\
&\iff (m_u, m'_u) \in (T_{u,u}(\varphi))^{\mathcal{A}}
\end{aligned}
$$

(2) $\varphi \equiv x = x$: $[\![x = x]\!]^{x,y,z}_{\mathcal{A}} = |\mathcal{A}|^3$, and hence $[\![x = x]\!]^{u,v}_{\mathcal{A}} = \top^{\mathcal{A}}$.

(3) $\varphi \equiv x = y$: This part is proved by $[\![x = y]\!]^{x,y,z}_{\mathcal{A}} = \triangle(|\mathcal{A}|) \times |\mathcal{A}|$. For example, $[\![x = y]\!]^{x,y}_{\mathcal{A}} = \triangle(|\mathcal{A}|) = 1^{\mathcal{A}}$ and $[\![x = y]\!]^{x,z}_{\mathcal{A}} = |\mathcal{A}|^2 = \top^{\mathcal{A}}$.

(4) $\varphi \equiv x\ a\ x$: This part is proved by $[\![x\ a\ x]\!]^{x,y,z}_{\mathcal{A}} = \{m \in |\mathcal{A}| \mid (m,m) \in a^{\mathcal{A}}\} \times |\mathcal{A}|^2$. For example,

$$
\begin{aligned}
(m_x, m_y) \in [\![x\ a\ x]\!]^{x,y}_{\mathcal{A}} &\iff \exists m_z.(m_x, m_y, m_z) \in \{m \in |\mathcal{A}| \mid (m,m) \in a^{\mathcal{A}}\} \times |\mathcal{A}|^2 \\
&\iff \exists m_z.(m_x, m_z) \in (a \cap 1)^{\mathcal{A}} \wedge (m_z, m_y) \in |\mathcal{A}|^2 \\
&\iff (m_x, m_y) \in (a \cap 1)^{\mathcal{A}} \cdot \top^{\mathcal{A}} \\
&\iff (m_x, m_y) \in ((a \cap 1) \cdot \top)^{\mathcal{A}}
\end{aligned}
$$

(5) $\varphi \equiv x\ a\ y$: This part is proved by $[\![x\ a\ y]\!]^{x,y,z}_{\mathcal{A}} = a^{\mathcal{A}} \times |\mathcal{A}|$. For example,

$$\begin{aligned}(m_x, m_z) \in [\![x\ a\ y]\!]^{x,z}_{\mathcal{A}} &\iff \exists m_y.(m_x, m_y, m_z) \in a^{\mathcal{A}} \times |\mathcal{A}| \\ &\iff \exists m_y.(m_x, m_y) \in a^{\mathcal{A}} \wedge (m_y, m_z) \in |\mathcal{A}|^2 \\ &\iff (m_x, m_y) \in (a \cdot \top)^{\mathcal{A}}\end{aligned}$$

$$\begin{aligned}(m_y, m_x) \in [\![x\ a\ y]\!]^{y,x}_{\mathcal{A}} &\iff \exists m_z.(m_x, m_y, m_z) \in a^{\mathcal{A}} \times |\mathcal{A}| \\ &\iff (m_x, m_y) \in a^{\mathcal{A}} \\ &\iff (m_y, m_x) \in (a^{\smile})^{\mathcal{A}}\end{aligned}$$

(6) $\varphi \equiv \exists z.\varphi$: This part is proved by $[\![\exists z.\varphi]\!]^{x,y,z}_{\mathcal{A}} = [\![\varphi]\!]^{x,y}_{\mathcal{A}} \times |\mathcal{A}|$. For example,

$$\begin{aligned}(m_x, m_z) \in [\![\exists z.\varphi]\!]^{x,z}_{\mathcal{A}} &\iff \exists m_y.(m_x, m_y, m_z) \in [\![\varphi]\!]^{x,y}_{\mathcal{A}} \times |\mathcal{A}| \\ &\iff \exists m_y.(m_x, m_y) \in [\![\varphi]\!]^{x,y}_{\mathcal{A}} \wedge (m_y, m_z) \in |\mathcal{A}|^2 \\ &\iff (m_x, m_z) \in [\![\varphi]\!]^{x,y}_{\mathcal{A}} \cdot \top^{\mathcal{A}} \\ &\iff (m_x, m_z) \in (T_{x,y}(\varphi) \cdot \top)^{\mathcal{A}} \qquad \text{(I.H.)}\end{aligned}$$

$$\begin{aligned}(m_x, m_y) \in [\![\exists z.\varphi]\!]^{x,y}_{\mathcal{A}} &\iff \exists m_z.(m_x, m_y, m_z) \in [\![\varphi]\!]^{x,y}_{\mathcal{A}} \times |\mathcal{A}| \\ &\iff \exists m_z.(m_x, m_z) \in [\![\varphi]\!]^{x,z}_{\mathcal{A}} \wedge (m_z, m_y) \in [\![\varphi]\!]^{z,y}_{\mathcal{A}} \\ &\iff (m_x, m_y) \in [\![\varphi]\!]^{x,z}_{\mathcal{A}} \cdot [\![\varphi]\!]^{z,y}_{\mathcal{A}} \\ &\iff (m_x, m_y) \in (T_{x,z}(\varphi) \cdot T_{z,y}(\varphi))^{\mathcal{A}} \qquad \text{(I.H.)}\end{aligned}$$

(7) $\varphi \equiv \neg\rho^{\{x,y\}}$: This part is proved by $[\![\neg\rho^{\{x,y\}}]\!]^{x,y,z}_{\mathcal{A}} = (|\mathcal{A}|^2 \setminus [\![\rho^{\{x,y\}}]\!]^{x,y}_{\mathcal{A}}) \times |\mathcal{A}|$. For example,

$$\begin{aligned}
(m_x, m_z) \in [\![\neg\rho^{\{x,y\}}]\!]^{x,z}_{\mathcal{A}} &\iff \exists m_y.(m_x, m_y, m_z) \in (|\mathcal{A}|^2 \setminus [\![\rho^{\{x,y\}}]\!]^{x,y}_{\mathcal{A}}) \times |\mathcal{A}| \\
&\iff \exists m_y.(m_x, m_y) \in (|\mathcal{A}|^2 \setminus [\![\rho^{\{x,y\}}]\!]^{x,y}_{\mathcal{A}}) \wedge (m_y, m_z) \in |\mathcal{A}|^2 \\
&\iff (m_x, m_z) \in (|\mathcal{A}|^2 \setminus [\![\rho^{\{x,y\}}]\!]^{x,y}_{\mathcal{A}}) \cdot \top^{\mathcal{A}} \\
&\iff (m_x, m_z) \in (T_{x,y}(\rho^{\{x,y\}})^- \cdot \top)^{\mathcal{A}} \qquad \text{(I.H.)}
\end{aligned}$$

(8) $\varphi \equiv \hat{\psi}^{\{x,y\}} \wedge \hat{\psi}^{\{y,z\}} \wedge \hat{\psi}^{\{z,x\}}$:

$$\begin{aligned}
&(m_x, m_y) \in [\![\hat{\psi}^{\{x,y\}} \wedge \hat{\psi}^{\{y,z\}} \wedge \hat{\psi}^{\{z,x\}}]\!]^{x,y}_{\mathcal{A}} \\
&\iff \exists m_z.(m_x, m_y, m_z) \in [\![\hat{\psi}^{\{x,y\}} \wedge \hat{\psi}^{\{y,z\}} \wedge \hat{\psi}^{\{z,x\}}]\!]^{x,y,z}_{\mathcal{A}} \\
&\iff \exists m_z.(m_x, m_y, m_z) \in [\![\hat{\psi}^{\{x,y\}}]\!]^{x,y}_{\mathcal{A}} \times |\mathcal{A}| \wedge (m_x, m_y, m_z) \in [\![\hat{\psi}^{\{y,z\}} \wedge \hat{\psi}^{\{z,x\}}]\!]^{x,y,z}_{\mathcal{A}} \\
&\iff (m_x, m_y) \in [\![\hat{\psi}^{\{x,y\}}_{\mathcal{A}}]\!]^{x,y} \wedge \exists m_z.(m_x, m_z, m_y) \in ([\![\hat{\psi}^{\{z,x\}}]\!]^{x,z}_{\mathcal{A}} \times |\mathcal{A}|) \cap (|\mathcal{A}| \times [\![\hat{\psi}^{\{y,z\}}]\!]^{z,y}_{\mathcal{A}}) \\
&\iff (m_x, m_y) \in [\![\hat{\psi}^{\{x,y\}}_{\mathcal{A}}]\!]^{x,y} \wedge (\exists m_z.(m_x, m_z) \in [\![\hat{\psi}^{\{z,x\}}]\!]^{x,z}_{\mathcal{A}} \wedge (m_z, m_y) \in [\![\hat{\psi}^{\{y,z\}}]\!]^{z,y}_{\mathcal{A}}) \\
&\iff (m_x, m_y) \in [\![\hat{\psi}^{\{x,y\}}_{\mathcal{A}}]\!]^{x,y} \wedge (m_x, m_y) \in [\![\hat{\psi}^{\{z,x\}}]\!]^{x,z}_{\mathcal{A}} \cdot [\![\hat{\psi}^{\{y,z\}}]\!]^{z,y}_{\mathcal{A}} \\
&\iff (m_x, m_y) \in [\![\hat{\psi}^{\{x,y\}}_{\mathcal{A}}]\!]^{x,y} \cap ([\![\hat{\psi}^{\{z,x\}}]\!]^{x,z}_{\mathcal{A}} \cdot [\![\hat{\psi}^{\{y,z\}}]\!]^{z,y}_{\mathcal{A}}) \\
&\iff (m_x, m_y) \in (T_{x,y}(\hat{\psi}^{\{x,y\}}) \cap (T_{x,z}(\hat{\psi}^{\{z,x\}}) \cdot T_{z,y}(\hat{\psi}^{\{y,z\}})))^{\mathcal{A}} \qquad \text{(I.H.)}
\end{aligned}$$

(9) $\varphi \equiv \hat{\psi}_1^{\{x,y\}} \wedge \hat{\psi}_2^{\{x,y\}}$:

$$\begin{aligned}
&(m_x, m_y) \in [\![\hat{\psi}_1^{\{x,y\}} \wedge \hat{\psi}_2^{\{x,y\}}]\!]^{x,y} \\
&\iff \exists m_z.(m_x, m_y, m_z) \in [\![\hat{\psi}_1^{\{x,y\}} \wedge \hat{\psi}_2^{\{x,y\}}]\!]^{x,y,z}_{\mathcal{A}} \\
&\iff \exists m_z.(m_x, m_y, m_z) \in ([\![\hat{\psi}_1^{\{x,y\}}]\!]^{x,y}_{\mathcal{A}} \cap [\![\hat{\psi}_2^{\{x,y\}}]\!]^{x,y}_{\mathcal{A}}) \times |\mathcal{A}| \\
&\iff (m_x, m_y) \in [\![\hat{\psi}_1^{\{x,y\}}]\!]^{x,y}_{\mathcal{A}} \wedge (m_x, m_y) \in [\![\hat{\psi}_2^{\{x,y\}}]\!]^{x,y}_{\mathcal{A}} \\
&\iff (m_x, m_y) \in (T_{x,y}(\hat{\psi}_1^{\{x,y\}}) \cap T_{x,y}(\hat{\psi}_2^{\{x,y\}}))^{\mathcal{A}} \qquad \text{(I.H.)}
\end{aligned}$$

(10) $\varphi \equiv \varphi_1 \vee \varphi_2$:

$$
\begin{aligned}
&(m_u, m_v) \in [\![\varphi_1 \vee \varphi_2]\!]_{\mathcal{A}}^{u,v} \\
&\iff \exists m_w.(m_u, m_v, m_w) \in [\![\varphi_1 \vee \varphi_2]\!]_{\mathcal{A}}^{u,v,w} \\
&\iff \exists m_w.(m_u, m_v, m_w) \in [\![\varphi_1]\!]_{\mathcal{A}}^{u,v,w}) \vee (m_u, m_v, m_w) \in [\![\varphi_2]\!]_{\mathcal{A}}^{u,v,w} \\
&\iff (\exists m_w.(m_u, m_v, m_w) \in [\![\varphi_1]\!]_{\mathcal{A}}^{u,v,w}) \vee (\exists m_w.(m_u, m_v, m_w) \in [\![\varphi_2]\!]_{\mathcal{A}}^{u,v,w}) \\
&\iff (m_u, m_v) \in [\![\varphi_1]\!]_{\mathcal{A}}^{u,v} \vee (m_u, m_v) \in [\![\varphi_2]\!]_{\mathcal{A}}^{u,v} \\
&\iff (m_u, m_v) \in [\![\varphi_1]\!]_{\mathcal{A}}^{u,v} \cup [\![\varphi_2]\!]_{\mathcal{A}}^{u,v} \\
&\iff (m_u, m_v) \in (T_{u,v}(\varphi_1) \cup T_{u,v}(\varphi_2))^{\mathcal{A}} \qquad \text{(I.H.)}
\end{aligned}
$$

Appendix B

Another Proof of Theorem 4.3.1(4)

The proof of this part is based on [43, Proposition 2.4]. We reduce from the membership problem for *nondeterministic linear-space bounded Turing machines*. It suffices to prove this part because, if a language L is NLINSPACE-hard under log-lin space reductions, then L is also PSPACE-hard under log space reductions [43, Lem 1.10.] (the class "CSL" in [43, Lem 1.10.] is equivalent to NLINSPACE [54]).

Let $M = (Q, A_M, \delta, q^I, q^a)$ be a nondeterministic linear-space bounded Turing machine and $s = a_1 \dots a_{n-1}$ be an input string, where (i) Q is a finite set of states; (ii) A_M is a finite set of alphabet, where A_M always contains the blank symbol $\sqcup$; (iii) $\delta : Q \times A_M \to \wp(Q \times A_M \times \{L, R\})$ is a transition function; (iv) $q^I \in Q$ is the initial state; and (v) $q^a \in Q$ is the acceptance state. We also require that once the machine enters its acceptance states, it never leaves it. M accepts a string s if the machine can reach an acceptance state q^a from the initial configuration (i.e, the header is at the leftest, the state is q^0, and the tape is $a_1 \dots a_n$) by finitely transitions. $L(M)$ denotes the set of all accepting strings.

We now construct a regular expression t^s_M such that $s \notin L(M)$ if and only if $L(t^s_M) \simeq_{\mathrm{p}} A^*$. Let A be the alphabet $A = \{\#\} \cup A_M \cup (Q \times A_M)$. $\frac{q}{a}$ denotes the character (q, a) in $Q \times A_M$. Moreover $A_{\setminus A'}$ denotes $A \setminus A'$ and the finite set $\{s_1, \dots, s_k\}$ denotes the regular expression $s_1 \cup \dots \cup s_k$. Let $f_M : A^3 \to \wp(A^3)$ (i.e., $f_M \subseteq A^3 \times A^3$) be the function denoting the transition on M. f_M is defined as the smallest binary relation on A^3 satisfying the follows:

(i) If $(q', a'_1, R) \in \delta(q, a_1)$, $(a'_1, \frac{q'}{a_2}, c_3) \in f_M(\frac{q}{a_1}, a_2, c_3)$;

(ii) If $(q', a'_1, L) \in \delta(q, a_1)$, $(a'_1, c_2, c_3) \in f_M(\frac{q}{a_1}, c_2, c_3)$;

(iii) If $(q', a'_2, R) \in \delta(q, a_2)$, $(c_1, a'_2, \frac{q'}{a_3}) \in f_M(c_1, \frac{q}{a_2}, a_3)$;

(iv) If $(q', a'_2, L) \in \delta(q, a_2)$, $(\frac{q'}{a_1}, a'_2, c_3) \in f_M(a_1, \frac{q}{a_2}, c_3)$;

(v) If $(q', a'_3, R) \in \delta(q, a_3)$, $(c_1, c_2, a'_3) \in f_M(c_1, c_2, \frac{q}{a_3})$;

(vi) If $(q', a'_3, L) \in \delta(q, a_3)$, $(c_1, \frac{q'}{a_2}, a'_3) \in f_M(c_1, a_2, \frac{q}{a_3})$;

(vii) If $(c_1, c_2, c_3) \in (\{\#\} \cup A_M)^3$, $(c_1, c_2, c_3) \in f_M(c_1, c_2, c_3)$.

Let $c_0^I \dots c_{n-1}^I = \#\frac{q^I}{a_1} a_2 \dots a_{n-1}$.

Then we define the following three regular expressions t_1, t_2 and t_3:

(1) (input error) $t_1 = (A_{\setminus\{c_0^I\}} \cup c_0^I(A_{\setminus\{c_1^I\}} \cup c_1^I(\dots(A_{\setminus\{c_{n-1}^I\}} \cup c_{n-1}^I A_{\setminus\{c_n^I\}})\dots)))A^*$;

(2) (acceptance error) $t_2 = (A_{\setminus\{q^a\}\times A_M})^*$;

(3) (transition error) $t_3 = \bigcup_{c_1,c_2,c_3\in A}(A_{\setminus\{q^a\}\times A_M})^* c_1c_2c_3A^{n-2}(A^3)_{\setminus f_M(c_1,c_2,c_3)}A^*$.

One can see the following hold:

(1) $A^* \setminus L(t_1) = \{s \in A^* \mid c_0^I \dots c_n^I$ is a prefix of $s\}$;

(2) $A^* \setminus L(t_2) = \{s \in A^* \mid s$ contains a character in $\{q^a\} \times A_M\}$;

(3) $c_0 \dots c_k \in A^* \setminus L(t_3)$ if any only if, for any i such that $n + i + 3 \leq k$ and $i \leq \min\{0 \leq j \leq k \mid c_j \in \{q^a\} \times A_M\}$, $c_{n+i+1}c_{n+i+2}c_{n+i+3} \in f_M(c_{i+1}c_{i+2}c_{i+3})$.

Let t_M^s be the regular expression defined as $t_M^s = t_1 \cup t_2 \cup t_3$. By the above three characterization for t_1, t_2, and t_3, $A^* \setminus L(t_M^s)$ is the set of the strings in the form of

$$\#c_1^0 \dots c_{n-1}^0 \#c_1^1 \dots c_{n-1}^1 \# \dots \#c_1^m \dots c_k^m s' \qquad (\heartsuit)$$

where (i) $c_1^0 \dots c_{n-1}^0 = c_1^I \dots c_{n-1}^I$ denotes the initial configuration; (ii) $c_1^i \dots c_{n-1}^i$ denotes a i-th configuration such that it is obtained from the $i-1$-th configuration denoted by $c_1^{i-1} \dots c_{n-1}^{i-1}$ for $i \leq k-1$; (iii) $c_k^m \in \{q^a\} \times A_M$, where $1 \leq k \leq n-1$ is a natural number; (iv) $c_1^m \dots c_k^m$ denotes the k-length prefix of a m-th configuration such that it is obtained from the $m-1$-th configuration denoted by $c_1^{m-1} \dots c_{n-1}^{m-1}$. (v) s' denotes an arbitrary string. Note that the language $L(t_M^s)$ is prefix-closed.

Then,

$$\begin{aligned} s \in L(M) &\iff L(t^s_M) \neq A^* && (\heartsuit) \\ &\iff L(t^s_M) \not\simeq_{\mathrm{p}} A^* && (L(t^s_M) \text{ is prefix-closed (Proposition 4.1.12(3)))} \end{aligned}$$

($\heartsuit$) is followed by that, if there is a string in $A^* \setminus L(t^s_M)$, an accepting run of s on M can be constructed from the string, and vice versa. By that t^s_M is reduced from s (and M, but M is fixed) in log-space, this part is proved. Moreover, by the length of t^s_M is linear of the length of s, this reduction is a log-lin space reduction. Therefore the p-universality problem for REGs is PSPACE-hard under log space reductions (more precisely, NLINSPACE-hard under log-lin space reductions).

Remark. The key modification from the reduction in [43, Prop.2.4] is (transition error). Let t'_3 be the following regular expression (see also "β_3" in [43, p.230]): $t'_3 = \bigcup_{c_1,c_2,c_3 \in A} A^* c_1 c_2 c_3 A^{n-2} (A^3)_{\setminus f_M(c_1,c_2,c_3)} A^*$. Then $c_0 \dots c_k \in A^* \setminus L(t'_3)$ if any only if, for any i such that $n + i + 3 \leq k$, $c_{n+i+1} c_{n+i+2} c_{n+i+3} \in f_M(c_{i+1} c_{i+2} c_{i+3})$. If t_3 is replaced by t'_3, $L(t^s_M)$ is not prefix-closed and the statement $L(t^s_M) \simeq_{\mathrm{p}} A^* \iff L(t^s_M) = A^*$ does not hold.

The above reduction also shows that the universality problem for regular expressions recognizing a prefix-closed language is NLINSPACE-complete under log-lin space reductions.

Bibliography

[1] Hajnal Andréka and D. A. Bredikhin. "The equational theory of union-free algebras of relations". In: *Algebra Universalis* 33.4 (1995), pp. 516–532. DOI: 10.1007/BF01225472.

[2] Hajnal Andréka, Szabolcs Mikulás, and István Németi. "The equational theory of Kleene lattices". In: *Theoretical Computer Science* 412.52 (2011), pp. 7099–7108. DOI: 10.1016/J.TCS.2011.09.024.

[3] Valentin Antimirov. "Partial derivatives of regular expressions and finite automaton constructions". In: *Theoretical Computer Science* 155.2 (1996), pp. 291–319. DOI: 10.1016/0304-3975(95)00182-4.

[4] Andrew Badr. "Hyper-Minimization in $O(n^2)$". In: *International Journal of Foundations of Computer Science* 20.04 (2009), pp. 735–746. DOI: 10.1142/S012905410900684X.

[5] Andrew Badr, Viliam Geffert, and Ian Shipman. "Hyper-minimizing minimized deterministic finite state automata". In: *RAIRO - Theoretical Informatics and Applications* 43.1 (2009), pp. 69–94. DOI: 10.1051/ita:2007061.

[6] Vince Bárány, Balder Ten Cate, and Luc Segoufin. "Guarded Negation". In: *Journal of the ACM* 62.3 (2015), pp. 1–26. DOI: 10.1145/2701414.

[7] Jean Berstel. "Sur la densité asymptotique de langages formels (in French)". In: *International Colloquium on Automata, Languages and Programming (ICALP, 1972)*. North-Holland, 1973, pp. 345–358.

[8] Jean Berstel, Dominique Perrin, and Christophe Reutenauer. *Codes and Automata*. Vol. 125. Cambridge University Press, 2009, p. 597.

[9] Patrick Blackburn, Johan van Benthem, and Frank Wolter. *Handbook of Modal Logic*. 1st ed. Vol. 3. Studies in Logic and Practical Reasoning. Elsevier, 2006, p. 1231.

[10] Andreas Blass, Yuri Gurevich, and Dexter Kozen. "A zero-one law for logic with a fixed-point operator". In: *Information and Control* 67.1-3 (1985), pp. 70–90. DOI: 10.1016/S0019-9958(85)80027-9.

[11] S. L. Bloom, Z. Ésik, and Gh. Stefanescu. "Notes on equational theories of relations". In: *Algebra Universalis* 33.1 (1995), pp. 98–126. DOI: 10.1007/BF01190768.

[12] Hans L. Bodlaender. "A partial k-arboretum of graphs with bounded treewidth". In: *Theoretical Computer Science* 209.1-2 (1998), pp. 1–45. DOI: 10.1016/S0304-3975(97)00228-4.

[13] Mikołaj Bojańczyk and Michał Pilipczuk. "Definability equals recognizability for graphs of bounded treewidth". In: *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science (LICS '16)*. ACM Press, 2016, pp. 407–416. DOI: 10.1145/2933575.2934508.

[14] Bela Bollobas. "The Diameter of Random Graphs". In: *Transactions of the American Mathematical Society* 267.1 (1981), p. 41. DOI: 10.2307/1998567.

[15] George S. Boolos, John P. Burgess, and Richard C. Jeffrey. *Computability and Logic*. 5th ed. Cambridge: Cambridge University Press, 2007, p. 350. DOI: 10.1017/CBO9780511804076.

[16] Egon Börger, Erich Grädel, and Yuri Gurevich. *The Classical Decision Problem*. Springer, 1997, p. 482.

[17] Paul Brunet and Damien Pous. "Petri automata". In: *Logical Methods in Computer Science* 13.3 (2017). DOI: 10.23638/LMCS-13(3:33)2017.

[18] Paul Brunet and Damien Pous. "Petri Automata for Kleene Allegories". In: *30th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS' 15)*. IEEE, 2015, pp. 68–79. DOI: 10.1109/LICS.2015.17.

[19] Janusz A. Brzozowski. "Derivatives of Regular Expressions". In: *Journal of the ACM* 11.4 (1964), pp. 481–494. DOI: 10.1145/321239.321249.

[20] Janusz A. Brzozowski, Jeffrey Shallit, and Zhi Xu. "Decision problems for convex languages". In: *Information and Computation* 209.3 (2011), pp. 353–367. DOI: 10.1016/j.ic.2010.11.009.

[21] Samuel R. Buss. "Algorithms for Boolean Formula Evaluation and for Tree Contraction". In: *Arithmetic, Proof Theory, and Computational Complexity*. Vol. 23. Oxford Logic Guides. 1993, pp. 96–115.

[22] Samuel R. Buss. "Boolean Formula Value Problem is in ALOGTIME". In: *Conference Proceedings of the Annual ACM Symposium on Theory of Computing (STOC' 87)*. ACM Press, 1987, pp. 123–131. DOI: 10.1145/28395.28409.

[23] Samuel R. Buss, Stepane A. Cook, Anshul Gupta, and Vijaya Ramachandran. "An Optimal Parallel Algorithm for Formula Evaluation". In: *SIAM Journal on Computing* 21.4 (1992), pp. 755–780. DOI: 10.1137/0221046.

[24] Alonzo Church. "A note on the Entscheidungsproblem". In: *The Journal of Symbolic Logic* 1.01 (1936), pp. 40–41. DOI: 10.2307/2269326.

[25] John H. Conway. *Regular Algebra and Finite Machines*. Chapman and Hall, 1971, p. 153.

[26] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. 3rd ed. The MIT Press, 2009, p. 1313.

[27] Anupam Das and Damien Pous. "A Cut-Free Cyclic Proof System for Kleene Algebra". In: *International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX 2017)*. Vol. 10501 LNCS. Springer, 2017, pp. 261–277. DOI: 10.1007/978-3-319-66902-1_16.

[28] Jules Desharnais and Georg Struth. "Internal axioms for domain semirings". In: *Science of Computer Programming* 76.3 (2011), pp. 181–203. DOI: 10.1016/J.SCICO.2010.05.007.

[29] Keith Ellul, Bryan Krawetz, Jeffrey Shallit, and Ming-wei Wang. "Regular expressions: new results and open problems". In: *Journal of Automata, Languages and Combinatorics* 9.2-3 (2004), pp. 233–256.

[30] David Eppstein. "Parallel recognition of series-parallel graphs". In: *Information and Computation* 98.1 (1992), pp. 41–55. DOI: 10.1016/0890-5401(92)90041-D.

[31] Ronald Fagin. "Generalized first-order spectra and polynomial-time recognizable sets". In: *Complexity of Computation*. Ed. by Richard M. Karp. Vol. 7. SIAM-AMS Proceedings, 1974, pp. 27–41.

[32] Peter J. Freyd and Andre Scedrov. *Categories*. Vol. 39. North-Holland Mathematical Library. North-Holland, 1990, pp. 2–192. DOI: 10.1016/S0924-6509(08)70048-5.

[33] Martin Fürer. "The complexity of the inequivalence problem for regular expressions with intersection". In: *International Colloquium on Automata, Languages,*

and Programming (ICALP '80). Vol. 85 LNCS. Springer, Berlin, Heidelberg, 1980, pp. 234–245. DOI: 10.1007/3-540-10003-2_74.

[34] Steven Givant. "The Calculus of Relations as a Foundation for Mathematics". In: *Journal of Automated Reasoning* 37.4 (2007), pp. 277–322. DOI: 10.1007/s10817-006-9062-x.

[35] Stefan Göller, Markus Lohrey, and Carsten Lutz. "PDL with intersection and converse: satisfiability and infinite-state model checking". In: *The Journal of Symbolic Logic* 74.01 (2009), pp. 279–314. DOI: 10.2178/jsl/1231082313.

[36] Etienne Grandjean. "Complexity of the first-order theory of almost all finite structures". In: *Information and Control* 57.2-3 (1983), pp. 180–204. DOI: 10.1016/S0019-9958(83)80043-6.

[37] Hermann Gruber and Markus Holzer. "Finite Automata, Digraph Connectivity, and Regular Expression Size". In: *the 35th International Colloquium on Automata, Languages and Programming (ICALP '08)*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 39–50. DOI: 10.1007/978-3-540-70583-3_4.

[38] Hermann Gruber and Markus Holzer. "From Finite Automata to Regular Expressions and Back—A Summary on Descriptional Complexity". In: *the Proceedings 14th International Conference on Automata and Formal Languages (AFL 2014)*. Vol. 151. 2014, pp. 25–48. DOI: 10.4204/EPTCS.151.2.

[39] David Harel, Dexter Kozen, and Jerzy Tiuryn. *Dynamic logic*. MIT Press, 2000, p. 459.

[40] Markus Holzer and Sebastian Jakobi. "From equivalence to almost-equivalence, and beyond - Minimizing automata with errors (extended abstract)". In: *Developments in Language Theory (DLT 2012)*. Vol. 7410 LNCS. Springer, Berlin, Heidelberg, 2012, pp. 190–201. DOI: 10.1007/978-3-642-31653-1_18.

[41] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. 3rd. Addison Wesley, 2006, p. 535.

[42] John E. Hopcroft and Jeffrey D. Ullman. *Formal languages and their relation to automata*. Addison-Wesley Longman Publishing Co., Inc., 1969, p. 262.

[43] Harry B. Hunt III, Daniel J. Rosenkrantz, and Thomas G. Szymanski. "On the equivalence, containment, and covering problems for the regular and context-free languages". In: *Journal of Computer and System Sciences* 12.2 (1976), pp. 222–268. DOI: 10.1016/S0022-0000(76)80038-4.

[44] Neil Immerman. *Descriptive Complexity*. Springer, 1999, p. 268. DOI: 10.1007/978-1-4612-0539-5.

[45] Neil Immerman. "Nondeterministic Space is Closed under Complementation". In: *SIAM Journal on Computing* 17.5 (1988), pp. 935–938. DOI: 10.1137/0217058.

[46] Neil D. Jones. "Space-bounded reducibility among combinatorial problems". In: *Journal of Computer and System Sciences* 11.1 (1975), pp. 68–85. DOI: 10.1016/S0022-0000(75)80050-X.

[47] A. S. Kahr, Edward F. Moore, and Hao Wang. "Entscheidungsproblem reduced to the AEA case". In: *Proceedings of the National Academy of Sciences* 48.3 (1962), pp. 365–377. DOI: 10.1073/pnas.48.3.365.

[48] Jui Yi Kao, Narad Rampersad, and Jeffrey Shallit. "On NFAs where all states are final, initial, or both". In: *Theoretical Computer Science* 410.47-49 (2009), pp. 5010–5021. DOI: 10.1016/j.tcs.2009.07.049.

[49] Richard M. Karp. "Reducibility among Combinatorial Problems". In: *Complexity of Computer Computations*. Springer, 1972, pp. 85–103. DOI: 10.1007/978-1-4684-2001-2_9.

[50] Stephen C. Kleene. "Representation of Events in Nerve Nets and Finite Automata". In: *Automata Studies*. Ed. by Claude Shanon and Jhon McCarthy. Princeton University Press, 1956, pp. 3–41.

[51] Dexter Kozen. "A completeness theorem for Kleene algebras and the algebra of regular events". In: *Proceedings Sixth Annual IEEE Symposium on Logic in Computer Science (LICS '91)*. IEEE Comput. Sco. Press, 1991, pp. 214–225. DOI: 10.1109/LICS.1991.151646.

[52] Dexter Kozen. "Kleene algebra with tests". In: *ACM Transactions on Programming Languages and Systems* 19.3 (1997), pp. 427–443. DOI: 10.1145/256167.256195.

[53] Dexter Kozen and Frederick Smith. "Kleene algebra with tests: Completeness and decidability". In: *International Workshop on Computer Science Logic* (1996).

[54] S.-Y. Kuroda. "Classes of languages and linear-bounded automata". In: *Information and Control* 7.2 (1964), pp. 207–223. DOI: 10.1016/S0019-9958(64)90120-2.

[55] Leonid Libkin. *Elements of Finite Model Theory*. Springer, 2012, p. 318. DOI: 10.1007/978-3-662-07003-1.

[56] Leopold Löwenheim. "Über Möglichkeiten im Relativkalkül". In: *Mathematische Annalen* 76.4 (1915), pp. 447–470. DOI: 10.1007/BF01458217.

[57] Carsten Lutz and Dirk Walther. "PDL with negation of atomic programs". In: *Journal of Applied Non-Classical Logics* 15.2 (2005), pp. 189–213. DOI: 10.3166/jancl.15.189-213.

[58] James F. Lynch. "Convergence laws for random words". In: *Australasian Journal of Combinatorics* 7 (1993), pp. 145–156.

[59] Nancy Lynch. "Log Space Recognition and Translation of Parenthesis Languages". In: *Journal of the ACM* 24.4 (1977), pp. 583–590. DOI: 10.1145/322033.322037.

[60] Roger C. Lyndon. "The Representation of Relation Algebras, II". In: *The Annals of Mathematics* 63.2 (1956), pp. 294–307. DOI: 10.2307/1969611.

[61] Roger C. Lyndon. "The Representation of Relational Algebras". In: *Annals of Mathematics* 51.3 (1950), pp. 707–729. DOI: 10.2307/1969375.

[62] Roger D. Maddux. *Relation Algebras*. Vol. 150. Studies in Logic and the Foundations of Mathematics. Elsevier, 2006, p. 758. DOI: 10.1016/S0049-237X(06)80023-6.

[63] Roger D. Maddux. "Undecidable semiassociative relation algebras". In: *The Journal of Symbolic Logic* 59.02 (1994), pp. 398–418. DOI: 10.2307/2275397.

[64] A. R. Meyer and M. J. Fischer. "Economy of description by automata, grammars, and formal systems". In: *12th Annual Symposium on Switching and Automata Theory (SWAT 1971)*. IEEE, 1971, pp. 188–191. DOI: 10.1109/SWAT.1971.11.

[65] David A. Mix Barrington, Neil Immerman, and Howard Straubing. "On uniformity within NC^1". In: *Journal of Computer and System Sciences* 41.3 (1990), pp. 274–306. DOI: 10.1016/0022-0000(90)90022-D.

[66] Donald Monk. "On representable relation algebras." In: *The Michigan Mathematical Journal* 11.3 (1964), pp. 207–210. DOI: 10.1307/mmj/1028999131.

[67] Marian Muresan. *A Concrete Approach to Classical Analysis*. CMS Books in Mathematics. Springer, 2009, p. 433. DOI: 10.1007/978-0-387-78933-0.

[68] Yoshiki Nakamura. "Partial Derivatives on Graphs for Kleene Allegories". In: *32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS '17)*. IEEE, 2017, pp. 1–12. DOI: 10.1109/LICS.2017.8005132.

[69] Yoshiki Nakamura. "The Almost Equivalence by Asymptotic Probabilities for Regular Languages and Its Computational Complexities". In: *Proceedings of the Seventh International Symposium on Games, Automata, Logics and Formal Verification (GandALF '16)*. Vol. 226. EPTCS. 2016, pp. 272–286. DOI: 10.4204/EPTCS.226.19.

[70] Yoshiki Nakamura. "The KAT-like Fragment of PDL with Intersection". In: *8th Conference: Non-Classical Logics. Theory and Applications*. 2016, pp. 96–100.

[71] Yoshiki Nakamura. "The Undecidability of FO3 and the Calculus of Relations with Just One Binary Relation". In: *Logic and Its Applications: 8th Indian Conference (ICLA '19)*. Springer, 2019, pp. 108–120. DOI: 10.1007/978-3-662-58771-3_11.

[72] Damien Pous. "On the Positive Calculus of Relations with Transitive Closure". In: *35th Symposium on Theoretical Aspects of Computer Science (STACS '18)*. Vol. 96. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018, 3:1–3:16. DOI: 10.4230/LIPICS.STACS.2018.3.

[73] V. N. Redko. "On defining relations for the algebra of regular events". In: *Ukrain. Mat. Z* (1964), 16:120–126.

[74] Neil Robertson and P. D. Seymour. "Graph minors. I. Excluding a forest". In: *Journal of Combinatorial Theory, Series B* 35.1 (1983), pp. 39–61. DOI: 10.1016/0095-8956(83)90079-5.

[75] Arto Salomaa. "Two Complete Axiom Systems for the Algebra of Regular Events". In: *Journal of the ACM* 13.1 (1966), pp. 158–169. DOI: 10.1145/321312.321326.

[76] Arto Salomaa and Matti Soittola. *Automata-Theoretic Aspects of Formal Power Series*. Springer, 1978, p. 88. DOI: 10.1007/978-1-4612-6264-0.

[77] Walter Savitch. "Relationships between nondeterministic and deterministic tape complexities". In: *Journal of Computer and System Sciences* 4.2 (1970), pp. 177–192. DOI: 10.1016/S0022-0000(70)80006-X.

[78] Luc Segoufin and Balder Cate. "Unary negation". In: *Logical Methods in Computer Science* 9.3 (2013). Ed. by Erich Grädel. DOI: 10.2168/LMCS-9(3:25)2013.

[79] Alan Selman. "Completeness of calculii for axiomatically defined classes of algebras". In: *Algebra Universalis* 2.1 (1972), pp. 20–32. DOI: 10.1007/BF02945004.

[80] Ryoma Sin'ya. "Zero-One Law for Regular Languages". PhD thesis. Tokyo Institute of Technology, 2016, p. 51.

[81] L. J. Stockmeyer and A. R. Meyer. "Word problems requiring exponential time (Preliminary Report)". In: *Proceedings of the fifth annual ACM symposium on Theory of computing (STOC '73)*. ACM, 1973, pp. 1–9. DOI: 10.1145/800125.804029.

[82] Robert Szelepcsényi. "The method of forcing for nondeterministic automata". In: *Acta Informatica* 26.3 (1988), pp. 279–284. DOI: 10.1007/BF00299636.

[83] Alfred Tarski. "Contributions to the theory of models. III". In: *Indagationes Mathematicae (Proceedings)* 58 (1955), pp. 56–64. DOI: 10.1016/S1385-7258(55)50009-6.

[84] Alfred Tarski. "On the Calculus of Relations". In: *The Journal of Symbolic Logic* 6.3 (1941), pp. 73–89. DOI: 10.2307/2268577.

[85] Alfred Tarski and Steven Givant. *A formalization of set theory without variables*. Vol. 41. Colloquium Publications. American Mathematical Society, 1987, p. 318.

[86] Ken Thompson. "Programming Techniques: Regular expression search algorithm". In: *Communications of the ACM* 11.6 (1968), pp. 419–422. DOI: 10.1145/363347.363387.