Kleene Algebra Completeness

Sabine Broda, Sílvia Cavadas, Nelma Moreira

e-mail: sbb@dcc.fc.up.pt, silviacavadas@gmail.com, nam@dcc.fc.up.pt

CMUP & DCC-FC, Universidade do Porto

Technical Report Series: DCC-2014-07

Version 1.0 July 2014

Departamento de Ciência de Computadores

Faculdade de Ciências da Universidade do Porto

Rua do Campo Alegre, 1021/1055,

4169-007 PORTO,

PORTUGAL

Tel: 220 402 900 Fax: 220 402 950

http://www.dcc.fc.up.pt/Pubs/

Abstract

This paper gives a new presentation of Kozen's proof of Kleene algebra completeness featured in his article *A completeness theorem for Kleene algebras and the algebra of regular events*. A few new variants are introduced, shortening the proof. Specifically, we directly construct an ε-free automaton to prove an equivalent to Kleene's representation theorem (implementing Glushkov's instead of Thompson's construction), and we bypass the use of minimal automata by directly implementing a Myhill-Nerode equivalence relation on the union of equivalent deterministic automata.

1 Introduction

A Kleene algebra is an algebraic structure intended to model the equational theory of regular languages. Several distinct axiomatizations serving that purpose exist in the literature, namely by Salomaa [7] and Kozen [4]. Kozen's axiomatization introduces two equational implications similar to Salomaa's inference rules but with the advantage of being sound over several natural interpretations.

This paper deals with Kozen's axiomatization, giving a new presentation of the proof of completeness featured in [4]. While we essentially follow Kozen's ideas, we introduce a few new variants, shortening the proof and avoiding the need for some technical considerations. Specifically, we implement Glushkov's instead of Thompson's construction to prove an equivalent to Kleene's representation theorem (thereby avoiding the need for implementing the ε-elimination during determinization, by directly constructing an ε-free automaton), and we bypass the use of minimal automata by directly implementing a Myhill-Nerode equivalence relation on the union of equivalent deterministic automata. In our presentation we also try to make more explicit some reasoning steps that we felt were somewhat obscure.

Braibant and Pous [1] formalized what is essentially Kozen's proof within the Coq proof assistant but used an improved version of Thompson's construction, the ε-follow automaton of Ilie and Yu [3]. Related work also includes a derivative based proof by Kozen [5] and completeness proofs for weak versions of Kozen's axioms that allow the ommission of one of the equational implications (see [6] and references therein).

The rest of this paper is divided into three sections. The first two give the background theory needed for the proof: the first section presents Kozen's axiomatization of Kleene algebra and some of its more relevant properties, while the second one recalls some important definitions and facts about regular languages, regular expressions and finite automata. We give particular emphasis to the connection between the combinatorial and the algebraic approach to automata, which is something that lies at the core of Kozen's proof but is not explicitly pointed out there. Finally, in the third section we prove completeness, pointing out where our approach differs from Kozen's.

2 Kleene Algebra

A Kleene algebra (KA) as defined in [4] is an algebraic structure $(\mathcal{K}, +, \cdot, ^*, 0, 1)$ such that $(\mathcal{K}, +, \cdot, 0, 1)$ is an idempotent semiring, i.e. satisfies axioms (1)-(9) below, and * satisfies axioms (10)-(13). The natural order $\leq$ in $(\mathcal{K}, +, \cdot, 0, 1)$ is defined by $a \leq b$ iff $a+b = b$. As usual, we will omit the operator $\cdot$ whenever it does not give rise to any ambiguity and use the following precedence over the operators:

$+ < \cdot < {}^*$.

(1)	$a + (b + c) = (a + b) + c$	(8)	$(a + b)c = ac + bc$
(2)	$a + b = b + a$	(9)	$0a = a0 = 0$
(3)	$a + 0 = 0 + a = a$	(10)	$1 + aa^* \leq a^*$
(4)	$a + a = a$	(11)	$1 + a^*a \leq a^*$
(5)	$a(bc) = (ab)c$	(12)	$b + ax \leq x \quad \rightarrow \quad a^*b \leq x$
(6)	$a1 = 1a = a$	(13)	$b + xa \leq x \quad \rightarrow \quad ba^* \leq x$
(7)	$a(b + c) = ab + ac$		

It is easy to check that $\leq$ is indeed a partial order. Reflexivity is simply the indempotence of $+$: for all $a \in \mathcal{K}$, $a + a = a$, ie $a \leq a$. Antisymmetry comes from the fact that, if $a \leq b$ and $b \leq a$, $b = a + b = b + a = a$, and transitivity from the fact that, if $a \leq b$ and $b \leq c$, i.e. $a + b = b$ and $b + c = c$, $a + c = a + (b + c) = (a + b) + c = b + c = c$, hence $a \leq c$. It is also monotone with respect to the three Kleene algebra operators. In fact, suppose that $a \leq b$, i.e. $a + b = b$. Then $(a + c) + (b + c) = (a + b) + c = b + c$, hence $a + b \leq b + c$, and $ac + bc = (a + b)c = bc$, hence $ac \leq bc$ (and analogously $ca \leq cb$). Also, using the monotonicity of $+$ and $\cdot$ plus axiom (10), $1 + ab^* \leq 1 + bb^* \leq b^*$, hence, by axiom (12), $a^* \leq b^*$. Two more important properties of $\leq$ are the fact that 0 is the least element for $\leq$, since $a + 0 = a$ for all $a \in \mathcal{K}$, and the following lemma.

Lemma 1. *In an idempotent semiring, $a + b \leq c \iff a \leq c$ and $b \leq c$.*

Proof. Suppose that $a+b \leq c$. From monotonicity and the fact that $0 \leq b$ follows that $a \leq a+b \leq c$; and similarly $b \leq c$. On the other hand, if $a \leq c$ and $b \leq c$, by monotonicity $a + b \leq c + c = c$. □

Axioms (10)-(13), the ones that rule the behaviour of *, are equivalent to the claim that, given elements $a, b \in \mathcal{K}$, a^*b is the least solution of $b + ax \leq x$ and ba^* the least solution of $b + xa \leq x$. Hence in a Kleene algebra a^* gets determined by the idempotent semiring structure as the least solution of $1 + ax \leq x$ (or $1 + xa \leq x$), which proves the following result.

Lemma 2. *Given an idempotent semiring, one can define at most one unary operation * that turns it into a Kleene algebra.*

The next lemma gives some basic equalities and inequalities that are true in every Kleene algebra.

Lemma 3. *For all a in a Kleene algebra, $1 \leq a^*$, $a \leq a^*$, $a^*a^* = a^*$, $(a^*)^* = a^*$, $1 + aa^* = a^*$, $1 + a^*a = a^*$, $0^* = 1$ and $1^* = 1$.*

Proof. Applying lemma 1 to axiom (10) yields $1 \leq a^*$ and $aa^* \leq a^*$. From these follows, using monotonicity, that $a \leq aa^* \leq a^*$. From $1 \leq a^*$ also follows that $a^* \leq a^*a^*$; to prove the converse inequality, note, using lemma 1, that $a^* + aa^* \leq a^*$, hence, applying axiom (12), $a^*a^* \leq a^*$. Therefore $a^*a^* = a^*$. Since $a \leq a^*$ for all a, $a^* \leq (a^*)^*$; on the other hand, applying axiom (12) to $1 + a^*a^* \leq a^*$ gives $(a^*)^* \leq a^*$. Hence $(a^*)^* = a^*$. To prove $1 + aa^* = a^*$ (the proof for $1 + a^*a = a^*$ is analogous), note that the inequality $1 + aa^* \leq a^*$ is simply axiom (10); from it follows, by monotonicity, that $1 + a(1 + aa^*) \leq 1 + aa^*$, hence, applying axiom (12), we have the converse inequality $a^* \leq 1 + aa^*$. Putting $a = 0$ in $1 + aa^* = a^*$ yields $0^* = 1 + 0 \cdot 0^* = 1$. We already know that $1 \leq 1^*$; to prove the converse inequality, note that $1 + 1 \cdot 1 \leq 1$ implies, by axiom (12), $1^* = 1^* \cdot 1 \leq 1$. Hence $1^* = 1$. □

Finally, we state a few important results that will be needed in the proof later on. Proofs for *i*, *ii* and *iii*, which we include for the sake of completeness, can be found in [4].

Lemma 4. *For all a and b in a Kleene algebra,*

i. $ax = xb \quad \rightarrow \quad a^*x = xb^*$;

ii. $(ab)^*a = a(ba)^*$;

iii. $(a+b)^* = a^*(ba^*)^*$;

iv. $(1+a)^* = a^*$.

Proof. i. Suppose $ax = xb$. From $ax \leq xb$ follows, by monotonicity, $axb^* \leq xbb^*$. By axiom (10) and distributivity, $x + xbb^* \leq xb^*$. Therefore, by monotonicity, $x + axb^* \leq x + xbb^* \leq xb^*$, and hence, by (12), $a^*x \leq xb^*$. The converse implication follows from a symmetric argument using (11) and (13).

ii. Follows from *i.* since $(ab)a = a(ba)$.

iii. By monotonicity and the previous lemma, $a^*(ba^*)^* \leq (a+b)^*((a+b)(a+b)^*)^* \leq (a+b)^*((a+b)^*)^* \leq (a+b)^*(a+b)^* \leq (a+b)^*$. The converse inequality follows from axiom (12) noting that $1 + (a+b)a^*(ba^*)^* = 1 + aa^*(ba^*)^* + ba^*(ba^*)^* \leq a^*(ba^*)^*$, since it can be seen, using monotonicity and the previous lemma, that $1 \leq a^*(ba^*)^*$, $aa^*(ba^*)^* \leq a^*(ba^*)^*$ and $ba^*(ba^*)^* \leq (ba^*)^* \leq a^*(ba^*)^*$.

iv. Using *iv.* we have $(1+a)^* = 1^*(a1^*)^* = 1(a1)^* = a^*$.

□

3 Regular Languages, Regular Expressions and Automata

Let Σ be a finite alphabet. The languages over Σ with union for $+$, language concatenation for $\cdot$, Kleene star for *, $\emptyset$ for 0 and $\{\varepsilon\}$ for 1 form a Kleene algebra. We denote by $\mathbf{Reg}_\Sigma$ the smallest subalgebra (subset which is closed for $+$, $\cdot$ and *) that contains 0, 1 and $\{\sigma\}$ for all $\sigma \in \Sigma$. Its elements are called the regular languages, and are precisely the languages that can be finitely generated from 0, 1, $\{\sigma\}$ ($\sigma \in \Sigma$) and operators $+$, $\cdot$, *. The regular languages are represented in the natural way by the regular expressions, which are the formal terms generated by the grammar

$$\alpha \quad \rightarrow \quad 0 \mid 1 \mid \sigma \mid \alpha + \alpha \mid \alpha \cdot \alpha \mid \alpha^* \quad (\sigma \in \Sigma).$$

Formally, given a regular expression α, the language $\mathcal{L}(\alpha)$ represented by α is inductively defined as follows,

$$\begin{array}{lcllcl} \mathcal{L}(\sigma) & = & \{\sigma\} & \mathcal{L}(\alpha+\beta) & = & \mathcal{L}(\alpha) \cup \mathcal{L}(\beta) \\ \mathcal{L}(0) & = & \emptyset & \mathcal{L}(\alpha\beta) & = & \mathcal{L}(\alpha)\mathcal{L}(\beta) \\ \mathcal{L}(1) & = & \{\varepsilon\} & \mathcal{L}(\alpha^*) & = & \mathcal{L}(\alpha)^*. \end{array}$$

The empty word operator ε on languages is defined by $\varepsilon(L) = 1$ if $\varepsilon \in L$ and $\varepsilon(L) = 0$ otherwise. An analogous operator is defined on regular expressions in the natural way by $\varepsilon(\alpha) = \varepsilon(\mathcal{L}(\alpha))$. An alternative inductive definition is

$$\begin{array}{lcllcl} \varepsilon(\sigma) & = & 0 & \varepsilon(\alpha+\beta) & = & \varepsilon(\alpha) + \varepsilon(\beta) \\ \varepsilon(0) & = & 0 & \varepsilon(\alpha\beta) & = & \varepsilon(\alpha)\varepsilon(\beta) \\ \varepsilon(1) & = & 1 & \varepsilon(\alpha^*) & = & 1. \end{array}$$

The regular languages are precisely the languages accepted by a finite automaton. There are three widely used classes of finite automata, all of which have the same power of expression despite their increasing generality: DFAs (deterministic finite automata), NFAs (nondeterministic finite automata) and ε-NFAs (nondeterministic finite automata with epsilon transitions). We recall that, considering a fixed alphabet Σ, a DFA is a tuple $\mathcal{A} = (Q, q_0, \delta, F)$ where Q is a finite set of states,

$q_0 \in Q$ the initial state, $F \subseteq Q$ a set of final states, and $\delta : Q \times \Sigma \longrightarrow Q$ the transition function (we say that in $\mathcal{A}$ there is a transition by σ from state q to state $\delta(q, \sigma)$). An NFA is a generalized DFA in which we allow the possibility of more than one initial state and of multiple (or none) transitions by the same symbol: instead of q_0 we consider a non-empty set $Q_0 \subseteq Q$ of initial states, and the transition function $\delta : Q \times \Sigma \longrightarrow \mathcal{P}(Q)$ now takes a state and symbol to a set of states. Finally, in an ε-NFA the transition function $\delta : Q \times \Sigma \cup \{\varepsilon\} \longrightarrow \mathcal{P}(Q)$ allows also for the possibility of making "spontaneous" transitions by the empty word ε. In each of these automata, we say that a word x can be read from state i to state j iff there is a sequence of transitions $i = q_1 \longrightarrow q_2 \longrightarrow \cdots \longrightarrow q_k = j$ in the automaton such that the concatenation of transition symbols (letters in Σ and/or the empty word) in the sequence is precisely x. The language accepted by a state q is the set of words that can be read from q to a final state; the language accepted by an automaton $\mathcal{A}$, denoted by $\mathcal{L}(\mathcal{A})$, is the set of words that can be read from an initial state to a final state. It is clear that DFAs are a particular case of NFAs and NFAs a particular case of ε-NFAs. To prove that in fact all three classes of automata have the same power of expression, there are standard constructions that allow to go from an ε-NFA to an NFA (epsilon elimination) and from an NFA to a DFA (subset construction) while preserving the accepted language. There are also a number of algorithms that convert from regular expressions to automata representing the same language (such as Thompson's, Brzozowski's, Glushkov's and Antimirov's algorithms) and the other way around (state elimination algorithm, MNY algorithm). For an overview of these topics we direct the reader to [**?**].

There is a natural correspondence between automata seen as tuples (Q, Q_0, δ, F) and the set of tuples (u, A, v) where A is an $Q \times Q$ matrix with entries in $\mathcal{P}(\{\varepsilon\} \cup \Sigma)$ and u and v are $Q \times 1$ vectors with entries in {0,1}: A, called the transition matrix, encodes Q and δ by interpreting its (p, q) entry as the set of transition symbols taking state p to state q, whereas u and v, called the initial states vector and final states vector, are the characteristic vectors of Q_0 and F. Note that the transition matrix is uniquely written as a sum $A = J + \sum_{\sigma \in \Sigma} \sigma A_\sigma$ where J and the A_σ are 0-1 matrices. The automaton is an NFA if J is the zero matrix, and a DFA if, besides being an NFA, u and all the rows in A_σ have exactly one entry equal to 1, in which case $\delta(q, \sigma)$ is the column index corresponding to the only 1 in the qth column of A_σ. Besides giving an alternative way to present an automaton, this new approach allows an algebraic characterization of its accepted language. It is easy to check that, considering the usual matrix product, the (p, q) entry of A^n is the set of words that can be read in (u, A, v) from state p to state q using n transitions. Thus, if we define A^* to be $I + A + A^2 + \cdots$, we have that the (p, q) entry of A^* is the set of words that can be read from state p to state q, and the language accepted by (u, A, v) is precisely $u^T A^* v$, the union of languages that can be read from an initial state to a final state.

Two automata, states or regular expressions are said to be language equivalent if they accept/represent the same language. Let $\mathcal{A} = (Q, \delta, F)$ be a deterministic transition system, i.e., a DFA without a defined start state. A Myhill-Nerode equivalence relation on $\mathcal{A}$ is an equivalence relation $\equiv$ on Q such that, if $p \equiv q$, then $p \in F \Leftrightarrow q \in F$ and $\delta(p, \sigma) \equiv \delta(q, \sigma)$ for all $\sigma \in \Sigma$. Given such an equivalence relation one can construct the quotient transition system $(Q/\equiv, \delta', \{[q] \mid q \in F\})$ where $\delta'([q], \sigma) = [\delta(q, \sigma)]$. In this quotient transition system, the language accepted by $[q]$ is precisely the language accepted by q in the original system. This shows in particular that two states related by a Myhill-Nerode equivalence relation accept the same language. The converse also holds, since the equivalence relation $p \equiv q$ iff p and q accept the same language is Myhill-Nerode.

4 Completeness of the KA Axioms

Kozen's completeness theorem states the completeness of the KA axioms for the equational theory of regular languages:

Theorem (Kozen)**.** *If two regular expressions represent the same language, their equality is provable from the KA axioms.*

The proof works with elements of $\mathcal{F}_\Sigma$, the free Kleene algebra on generators Σ, which is constructed taking the quotient of the regular expressions modulo provable equivalence by the KA axioms. One checks that this quotient is indeed a Kleene algebra with the operations induced by the syntactic operations in the regular expressions. Furthermore, since all regular expressions in the same equivalence class represent the same language, it makes sense to talk about the language represented by an element $\alpha \in \mathcal{F}_\Sigma$, denoted by $\mathcal{L}(\alpha)$. This gives a natural Kleene algebra surjective homomorphism between $\mathcal{F}_\Sigma$ and $\mathbf{Reg}_\Sigma$. What the theorem states is that this homomorphism is in fact also injective.

The concept that plays a central role in the proof is that of a finite automaton over $\mathcal{F}_\Sigma$, an object analogous to the usual finite automata that is able to recognize a certain element of $\mathcal{F}_\Sigma$. The definition of automata in this context, and more generally in the context of an arbitrary Kleene algebra, is motivated by the algebraic approach to automata theory presented above, and relies crucially on some structure on the matrices over that Kleene algebra.

4.1 Matrices over a Kleene Algebra

It is easy to check that, if $\mathcal{K}$ is a Kleene algebra, the usual sum and product between matrices over $\mathcal{K}$ satisfy the axioms of an idempotent semiring (with the zero matrix in the role of 0 and the identity matrix in the role of 1) as long as the operations are defined between the operands. In particular, $\mathcal{M}(n, \mathcal{K})$ is an idempotent semiring for every $n \in \mathbb{N}$. If $\mathcal{K}$ is the set of languages over a certain alphabet, one can aditionally define a unary * operation for square matrices, as done above, by $M^* = I + M + M^2 + \cdots$. It is straightforward to check that this definition satisfies the * Kleene algebra axioms, turning $\mathcal{M}(n, \mathcal{K})$ into a Kleene algebra. In the case of a general $\mathcal{K}$, such definition does not apply, but an alternative inductive definition for * is possible (which, by Lemma 2, coincides with the above definition when $\mathcal{K}$ is the set of languages over some alphabet). We part slightly from the definition given by Kozen, preferring the more symmetric approach used by Conway in [2]. For 1×1 matrices, we simply define $[a]^* = [a^*]$, turning $\mathcal{M}(1, \mathcal{K})$ into a Kleene algebra isomorphic to $\mathcal{K}$. Having defined * for matrices of order up to n, we partition each matrix M of order $n+1$ into submatrices

$$M = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$$

so that A and D are square and define

$$M^* = \left[\begin{array}{c|c} (A + BD^*C)^* & A^*B(D + CA^*B)^* \\ \hline D^*C(A + BD^*C)^* & (D + CA^*B)^* \end{array}\right].$$

We then check that this definition satisfies the * KA axioms, so that $\mathcal{M}(n+1, \mathcal{K})$ is a Kleene algebra. It follows, by Lemma 2, that the definition of * does not depend on the partitions chosen. By induction hypothesis, the operator * satisfies the KA axioms for matrices of dimension less than $n+1$. To show $I + MM^* \leq M^*$ (the proof for $I + M^*M \leq M^*$ is analogous), note that, setting

$E = A + BD^*C$ and $F = D + CA^*B$, this inequality reduces to the four inequalities

$$\begin{aligned}
I + AE^* + BD^*CE^* \leq E^* &\Leftrightarrow I + EE^* \leq E^* \\
AA^*BF^* + BF^* \leq A^*BF^* &\Leftrightarrow (AA^* + I)BF^* \leq A^*BF^* \\
CE^* + DD^*CE^* \leq D^*CE^* &\Leftrightarrow (I + DD^*)CE^* \leq D^*CE^* \\
I + CA^*BF^* + DF^* \leq F^* &\Leftrightarrow I + FF^* \leq F^*
\end{aligned}$$

which follow from axiom (10) and monotonicity. We now show that $Y + MX \leq X \rightarrow M^*Y \leq X$ (the proof for $Y + XM \leq X \rightarrow YM^* \leq X$ is analogous). Let

$$X = \left[\begin{array}{c|c} X_1 & X_2 \\ \hline X_3 & X_4 \end{array}\right] \quad \text{and} \quad Y = \left[\begin{array}{c|c} Y_1 & Y_2 \\ \hline Y_3 & Y_4 \end{array}\right]$$

and assume that $Y + MX \leq X$, that is,

$$\begin{aligned}
Y_1 + AX_1 + BX_3 \leq X_1 \qquad & Y_3 + CX_1 + DX_3 \leq X_3 \\
Y_2 + AX_2 + BX_4 \leq X_2 \qquad & Y_4 + CX_2 + DX_4 \leq X_4.
\end{aligned}$$

Then

$$\begin{aligned}
AX_1 + BX_3 \leq X_1 \qquad & CX_1 + DX_3 \leq X_3 \\
AX_2 + BX_4 \leq X_2 \qquad & CX_2 + DX_4 \leq X_4,
\end{aligned}$$

hence

$$\begin{aligned}
A^*BX_3 \leq X_1 \qquad & D^*CX_1 \leq X_3 \\
A^*BX_4 \leq X_2 \qquad & D^*CX_2 \leq X_4,
\end{aligned}$$

and thus, substituting in the first set of equations,

$$\begin{aligned}
Y_1 + AX_1 + BD^*CX_1 \leq X_1 \qquad & Y_3 + DX_3 + CA^*BX_3 \leq X_3 \\
Y_2 + AX_2 + BD^*CX_2 \leq X_2 \qquad & Y_4 + DX_4 + CA^*BX_4 \leq X_4,
\end{aligned}$$

hence

$$\begin{aligned}
(A + BD^*C)^*Y_1 \leq X_1 \Leftrightarrow E^*Y_1 \leq X_1 \qquad & (D + CA^*B)^*Y_3 \leq X_3 \Leftrightarrow F^*Y_3 \leq X_3 \\
(A + BD^*C)^*Y_2 \leq X_2 \Leftrightarrow E^*Y_2 \leq X_2 \qquad & (D + CA^*B)^*Y_4 \leq X_4 \Leftrightarrow F^*Y_4 \leq X_4
\end{aligned}$$

and, substituting in the second set of equations,

$$\begin{aligned}
AX_1 + BF^*Y_3 \leq X_1 \qquad & DX_3 + CE^*Y_1 \leq X_3 \\
AX_2 + BF^*Y_4 \leq X_2 \qquad & DX_4 + CE^*Y_2 \leq X_4
\end{aligned}$$

hence

$$\begin{aligned}
A^*BF^*Y_3 \leq X_1 \qquad & D^*CE^*Y_1 \leq X_3 \\
A^*BF^*Y_4 \leq X_2 \qquad & D^*CE^*Y_2 \leq X_4
\end{aligned}$$

From the fifth and the last sets of equations follows

$$\begin{aligned}
E^*Y_1 + A^*BF^*Y_3 \leq X_1 \qquad & D^*CE^*Y_1 + F^*Y_3 \leq X_3 \\
E^*Y_2 + A^*BF^*Y_4 \leq X_2 \qquad & D^*CE^*Y_2 + F^*Y_4 \leq X_4,
\end{aligned}$$

which is precisely $M^*Y \leq X$.

4.2 Finite Automata

A finite automaton over a Kleene algebra $\mathcal{K}$ is a triple $\mathcal{A} = (u, A, v)$ where $u, v \in \{0,1\}^n$ and $A \in \mathcal{M}(n, \mathcal{K})$ for some $n \in \mathbb{N}$. By analogy with the algebraic definition of automaton given in Section 3, we say that the matrix indices are the automaton states, and call A the transition matrix, u the initial states vector and v the final states vector. The element of $\mathcal{K}$ accepted by $\mathcal{A}$ is $u^T A^* v$.

We now consider the case where $\mathcal{K} = \mathcal{F}_\Sigma$. The automaton $\mathcal{A}$ is said to be simple if A can be expressed as a sum $A = J + \sum_{\sigma\in\Sigma} \sigma A_\sigma$ where J and the A_σ are 0-1 matrices. Note that, when it exists, this decomposition is unique. $\mathcal{A}$ is said to be ε-free is J is the zero matrix, and deterministic if it is simple, ε-free, and u and all rows of A_σ have exactly one 1.

A simple automaton $\mathcal{A} = (u, A, v)$ encodes in a natural way an ε-NFA $\mathcal{A}' = (u', A', v')$ where u', A' and v' are obtained from u, A and v by simply replacing each element of $\mathcal{F}_\Sigma$ by the regular language is represents. Since this correspondence is a homomorphism between $\mathcal{F}_\Sigma$ and $\mathbf{Reg}_\Sigma$, $\mathcal{L}(\mathcal{A}') = u'^T A'^* v' = \mathcal{L}(u^T A^* v)$: the language accepted by $\mathcal{A}'$ is the language represented by $u^T A^* v$. It is also clear that $\mathcal{A}'$ is an NFA exactly when $\mathcal{A}$ is ε-free, and a DFA when $\mathcal{A}$ is deterministic.

We start by proving the equivalent of Kleene's representation theorem for free regular expressions: that every element of $\mathcal{F}_\Sigma$ is accepted by some automaton. In his article, Kozen does so by simulating Thompson's construction to build a simple automaton that accepts a given $\mathcal{F}_\Sigma$ element. He then gets rid of ε-transitions and mimics the subset construction to obtain an equivalent deterministic automaton. Here we will, following the same ideas, directly construct an ε-free simple automata by emulating Glushkov's construction (see [**?**] for a description of Glushkov's automaton).

Lemma 5. *For every free regular expression α there is an ε-free simple automaton $\mathcal{A} = (u, A, v)$ such that $u^T A^* v = \alpha$.*

Proof. We proceed by structural induction. For each free regular expression α we will obtain a corresponding ε-free simple automaton of the form

$$G_\alpha = \left(\left[\begin{array}{c} 1 \\ \hline 0 \end{array}\right], \left[\begin{array}{c|c} 0 & a \\ \hline 0 & A \end{array}\right], \left[\begin{array}{c} \varepsilon(\alpha) \\ \hline v \end{array}\right] \right)$$

where A is a square matrix, a a row vector and the partitions are compatible with the matrix operations. Each automaton will thus admit a single start state to which no transitions arrive and which is a final state iff $\varepsilon(\alpha) = 1$. For the base case, we present automata satisfying these conditions that accept 0, 1 and a for all $a \in \Sigma$:

$$G_0 = \left(\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right), \quad G_1 = \left(\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right), \quad G_a = \left(\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 & a \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right).$$

Suppose now that α and β are accepted by the automata

$$G_\alpha = \left(\left[\begin{array}{c} 1 \\ \hline 0 \end{array}\right], \left[\begin{array}{c|c} 0 & a \\ \hline 0 & A \end{array}\right], \left[\begin{array}{c} \varepsilon(\alpha) \\ \hline v \end{array}\right] \right), \quad G_\beta = \left(\left[\begin{array}{c} 1 \\ \hline 0 \end{array}\right], \left[\begin{array}{c|c} 0 & b \\ \hline 0 & B \end{array}\right], \left[\begin{array}{c} \varepsilon(\beta) \\ \hline t \end{array}\right] \right)$$

so that

$$\alpha = \left[\begin{array}{c|c} 1 & 0 \end{array}\right] \left[\begin{array}{c|c} 0 & a \\ \hline 0 & A \end{array}\right]^* \left[\begin{array}{c} \varepsilon(\alpha) \\ \hline v \end{array}\right] = \left[\begin{array}{c|c} 1 & 0 \end{array}\right] \left[\begin{array}{c|c} 1 & aA^* \\ \hline 0 & A^* \end{array}\right] \left[\begin{array}{c} \varepsilon(\alpha) \\ \hline v \end{array}\right] = \varepsilon(\alpha) + aA^*v$$

$$\beta = \left[\begin{array}{c|c} 1 & 0 \end{array}\right] \left[\begin{array}{c|c} 0 & b \\ \hline 0 & B \end{array}\right]^* \left[\begin{array}{c} \varepsilon(\beta) \\ \hline t \end{array}\right] = \left[\begin{array}{c|c} 1 & 0 \end{array}\right] \left[\begin{array}{c|c} 1 & bB^* \\ \hline 0 & B^* \end{array}\right] \left[\begin{array}{c} \varepsilon(\beta) \\ \hline t \end{array}\right] = \varepsilon(\beta) + bB^*t.$$

Then $\alpha + \beta$ is accepted by the automaton

$$G_{\alpha+\beta} = \left(\left[\begin{array}{c} 1 \\ \hline 0 \\ \hline 0 \end{array}\right], \left[\begin{array}{c|c|c} 0 & a & b \\ \hline 0 & A & 0 \\ \hline 0 & 0 & B \end{array}\right], \left[\begin{array}{c} \varepsilon(\alpha+\beta) \\ \hline v \\ \hline t \end{array}\right] \right)$$

because

$$\begin{aligned}
&\left[\begin{array}{c|c|c} 1 & 0 & 0 \end{array}\right] \left[\begin{array}{c|c|c} 0 & a & b \\ \hline 0 & A & 0 \\ \hline 0 & 0 & B \end{array}\right]^* \left[\begin{array}{c} \varepsilon(\alpha+\beta) \\ \hline v \\ \hline t \end{array}\right] \\
&= \left[\begin{array}{c|c|c} 1 & 0 & 0 \end{array}\right] \left[\begin{array}{c|c|c} 1 & aA^* & bB^* \\ \hline 0 & A^* & 0 \\ \hline 0 & 0 & B^* \end{array}\right] \left[\begin{array}{c} \varepsilon(\alpha)+\varepsilon(\beta) \\ \hline v \\ \hline t \end{array}\right] \\
&= \varepsilon(\alpha) + \varepsilon(\beta) + aA^*v + bB^*t = \alpha + \beta;
\end{aligned}$$

$\alpha\beta$ is accepted by the automaton

$$G_{\alpha\beta} = \left(\left[\begin{array}{c} 1 \\ \hline 0 \\ \hline 0 \end{array}\right], \left[\begin{array}{c|c|c} 0 & a & \varepsilon(\alpha)b \\ \hline 0 & A & vb \\ \hline 0 & 0 & B \end{array}\right], \left[\begin{array}{c} \varepsilon(\alpha\beta) \\ \hline \varepsilon(\beta)v \\ \hline t \end{array}\right] \right)$$

because

$$\begin{aligned}
&\left[\begin{array}{c|c|c} 1 & 0 & 0 \end{array}\right] \left[\begin{array}{c|c|c} 0 & a & \varepsilon(\alpha)b \\ \hline 0 & A & vb \\ \hline 0 & 0 & B \end{array}\right]^* \left[\begin{array}{c} \varepsilon(\alpha\beta) \\ \hline \varepsilon(\beta)v \\ \hline t \end{array}\right] \\
&= \left[\begin{array}{c|c|c} 1 & 0 & 0 \end{array}\right] \left[\begin{array}{c|c|c} 1 & aA^* & aA^*vbB^* + \varepsilon(\alpha)bB^* \\ \hline 0 & A^* & A^*vbB^* \\ \hline 0 & 0 & B^* \end{array}\right] \left[\begin{array}{c} \varepsilon(\alpha)\varepsilon(\beta) \\ \hline \varepsilon(\beta)v \\ \hline t \end{array}\right] \\
&= \varepsilon(\alpha)\varepsilon(\beta) + aA^*\varepsilon(\beta)v + aA^*vbB^*t + \varepsilon(\alpha)bB^*t \\
&= \varepsilon(\alpha)\varepsilon(\beta) + (aA^*v)\varepsilon(\beta) + (aA^*v)(bB^*t) + \varepsilon(\alpha)(bB^*t) \\
&= (\varepsilon(\alpha) + aA^*v)(\varepsilon(\beta) + bB^*t) = \alpha\beta;
\end{aligned}$$

and α^* is accepted by the automaton

$$G_{\alpha^*} = \left(\left[\begin{array}{c} 1 \\ \hline 0 \end{array}\right], \left[\begin{array}{c|c} 0 & a \\ \hline 0 & A + va \end{array}\right], \left[\begin{array}{c} 1 \\ \hline v \end{array}\right] \right)$$

because, using Lemma 3 and Lemma 4,

$$\begin{aligned}
&\left[\begin{array}{c|c} 1 & 0 \end{array}\right] \left[\begin{array}{c|c} 0 & a \\ \hline 0 & A + va \end{array}\right]^* \left[\begin{array}{c} 1 \\ \hline v \end{array}\right] \\
&= \left[\begin{array}{c|c} 1 & 0 \end{array}\right] \left[\begin{array}{c|c} 1 & a(A+va)^* \\ \hline 0 & (A+va)^* \end{array}\right] \left[\begin{array}{c} 1 \\ \hline v \end{array}\right] \\
&= 1 + a(A+va)^*v \\
&= 1 + aA^*(vaA^*)^*v \\
&= 1 + (aA^*v)(aA^*v)^* = (aA^*v)^* = (\varepsilon(\alpha) + aA^*v)^* = \alpha^*.
\end{aligned}$$

□

The next step is to prove that for each ε-free simple automaton there is a deterministic automaton that accepts the same element of $\mathcal{F}_\Sigma$. We reproduce Kozen's proof, which works by implementing the subset construction algebraically.

Lemma 6. *For every simple ε-free automaton (u, A, v) there is a deterministic automaton $(\hat{u}, \hat{A}, \hat{v})$ such that $u^T A^* v = \hat{u}^T \hat{A}^* \hat{v}$.*

Proof. Let (u, A, v), with $A = \sum_{\sigma\in\Sigma} \sigma A_\sigma$, be a simple ε-free automaton with states Q, where Q is of size n. We identify each element $s \in \mathcal{P}(Q)$ with its characteristic $\{0,1\}^n$ vector, so that in the determinized automaton each state s makes a transition by σ to $s^T A_\sigma$. The transformation between states in Q and states in $\mathcal{P}(Q)$ will be codified by the 0-1 $\mathcal{P}(Q) \times Q$ matrix X whose sth row is s^T. Let e_s be the $\mathcal{P}(Q) \times 1$ vector with 1 in position s and 0 elsewhere. We define $\hat{A}_\sigma$ as the $\mathcal{P}(Q) \times \mathcal{P}(Q)$ matrix whose sth row is $e^T_{s^T A_\sigma}$, and set $\hat{A} = \sum_{\sigma\in\Sigma} \sigma \hat{A}_\sigma$. We then have the relation $XA = \hat{A}X$ because, for each $s \in \mathcal{P}(Q)$,

$$e_s^T XA = s^T A = \sum_{\sigma\in\Sigma} \sigma(s^T A_\sigma) = \sum_{\sigma\in\Sigma} \sigma(e^T_{s^T A_\sigma} X) = \sum_{\sigma\in\Sigma} \sigma(e_s^T \hat{A}_\sigma X) = e_s^T \hat{A} X.$$

If we aditionally define $\hat{u} = e_u$ and $\hat{v} = Xv$ we have that the automaton $(\hat{u}, \hat{A}, \hat{v})$ is deterministic and also accepts the same $\mathcal{F}_\Sigma$ element as (u, A, v): the proof rests on the fact that, by Lemma 3, $XA = \hat{A}X$ implies $XA^* = \hat{A}^* X$, hence

$$\hat{u}^T \hat{A}^* \hat{v} = e_u^T \hat{A}^* X v = e_u^T X A^* v = u^T A^* v.$$

□

From the two previous lemmas follows that for every $\alpha \in \mathcal{F}_\Sigma$ there is a deterministic finite automaton $\mathcal{A} = (u, A, v)$ such that $u^T A^* v = \alpha$.

4.3 Myhill-Nerode Equivalence Relations and Completeness

Suppose now that α and β belonging to $\mathcal{F}_\Sigma$ are language equivalent. We already know that there are deterministic automata $\mathcal{A} = (u, A, v)$ and $\mathcal{B} = (s, B, t)$ accepting α and β, i.e. such that $u^T A^* v = \alpha$ and $s^T B^* t = \beta$. We will show that α and β are the same, thereby proving completeness, by algebraically implementing the notion of a Myhill-Nerode equivalence relation on the union of states of $\mathcal{A}$ and $\mathcal{B}$ and showing that in the resulting quotient transition system the class corresponding to the initial states of $\mathcal{A}$ and $\mathcal{B}$ accepts the same element of $\mathcal{F}_\Sigma$ as both the original automata. Let $\mathcal{A}' = (Q_1, \delta_1, a_0, F_1)$ and $\mathcal{B}' = (Q_2, \delta_2, b_0, F_2)$ be the corresponding DFAs. The language accepted by both $\mathcal{A}'$ and $\mathcal{B}'$ is $\mathcal{L}(\alpha) = \mathcal{L}(\beta)$. Put $Q = Q_1 \cup Q_2$, where we can assume that Q_1 and Q_2 are disjoint, $\delta = \delta_1 \cup \delta_2$ and $F = F_1 \cup F_2$. Since a_0 and b_0 are language equivalent, we know that there is a Myhill-Nerode equivalence relation $\equiv$ on Q such that $a_0 \equiv b_0$. Writing $A = \sum_{\sigma\in\Sigma} \sigma A_\sigma$ and $B = \sum_{\sigma\in\Sigma} \sigma B_\sigma$, consider

$$M_\sigma = \left[\begin{array}{c|c} A_\sigma & 0 \\ \hline 0 & B_\sigma \end{array}\right], \quad M = \sum_{\sigma\in\Sigma} \sigma M_\sigma \quad \text{and} \quad y = \left[\begin{array}{c} v \\ t \end{array}\right].$$

We now algebraically implement the quotient by $\equiv$. Let Y be the $Q \times Q/\equiv$ matrix whose $[q]$th column is the characteristic vector of $[q]$, where $[q]$ is the equivalence class of the state q. Note that the partition $Q = Q_1 \cup Q_2$ induces a natural partition

$$Y = \left[\begin{array}{c} Y_A \\ Y_B \end{array}\right].$$

Let e_q be the $Q \times 1$ vector with 1 in position q and 0 otherwise, and $e_{[q]}$ the $Q/\equiv \times 1$ vector with 1 in position $[q]$ and 0 otherwise. Note that, by the definition of Y, $e_q^T Y = e_{[q]}^T$. For each $\sigma \in \Sigma$ we define $\overline{M}_\sigma$ as the $Q/\equiv \times Q/\equiv$ matrix whose $[q]$th row is $e^T_{[\delta(q,\sigma)]}$; this is well defined since $\equiv$ is Myhill-Nerode. Put $\overline{M} = \sum_{\sigma\in\Sigma} \sigma \overline{M}_\sigma$ and let $\overline{y}$ be the $Q/\equiv \times 1$ vector whose $[q]$th entry is equal to the qth entry of y; once again this is well defined because $\equiv$ is Myhill-Nerode. We then have that $Y\overline{y} = y$ and $MY = Y\overline{M}$, since, for any $q \in Q$,

$$e_q^T MY = \sum_{\sigma\in\Sigma} \sigma e_q^T M_\sigma Y = \sum_{\sigma\in\Sigma} \sigma e^T_{\delta(q,\sigma)} Y = \sum_{\sigma\in\Sigma} \sigma e^T_{[\delta(q,\sigma)]} = \sum_{\sigma\in\Sigma} \sigma e^T_{[q]} \overline{M}_\sigma = \sum_{\sigma\in\Sigma} \sigma e_q^T Y \overline{M}_\sigma = e_q^T Y \overline{M}.$$

Hence

$$\left[\begin{array}{c} Y_A\overline{y} \\ Y_B\overline{y} \end{array}\right] = \left[\begin{array}{c} v \\ t \end{array}\right]$$

and, by Lemma 4,

$$M^*Y = Y\overline{M}^* \Leftrightarrow \left[\begin{array}{c} A^*Y_A \\ B^*Y_B \end{array}\right] = \left[\begin{array}{c} Y_A\overline{M}^* \\ Y_B\overline{M}^* \end{array}\right].$$

Hence, noting that $u = e_{a_0}$ and $s = e_{b_0}$,

$$\begin{aligned} \alpha = u^T A^* v = e^T_{a_0} A^* Y_A \overline{y} = e^T_{a_0} Y_A \overline{M}^* \overline{y} = e^T_{[a_0]} \overline{M}^* \overline{y} \\ = e^T_{[b_0]} \overline{M}^* \overline{y} = e^T_{b_0} Y_B \overline{M}^* \overline{y} = e^T_{b_0} B^* Y_B \overline{y} = s^T B^* t = \beta, \end{aligned}$$

which concludes the proof. □

References

[1] Thomas Braibant and Damien Pous. Deciding Kleene algebras in Coq. *Logical Methods in Computer Science*, 8(1), 2012.

[2] John H. Conway. *Regular algebra and finite machines.* William Clowes& Sons, Great Britain, 1971.

[3] L. Ilie and S. Yu. Follow automata. *Inf. Comput.*, 186(1):140–162, 2003.

[4] D. C. Kozen. A completeness theorem for Kleene algebras and the algebra of regular events. *Infor. and Comput.*, 110(2):366–390, 05 1994.

[5] Dexter Kozen. Myhill-nerode relations on automatic systems and the completeness of kleene algebra. In Afonso Ferreira and Horst Reichel, editors, *18th STACS 2001*, volume 2010 of *LNCS*, pages 27–38. Springer, 2001.

[6] Dexter Kozen and Alexandra Silva. Left-handed completeness. In Wolfram Kahl and Timothy G. Griffin, editors, *13th RAMiCS 2012*, volume 7560 of *LNCS*, pages 162–178. Springer, 2012.

[7] A. Salomaa. Two complete axiom systems for the algebra of regular events. *Journal of the Association for Computing Machinery*, 13(1):158–169, 1966.