Introduction to Kleene Algebra	**Lecture 8**
CS786 Spring 2002	**Tuesday, February 19, 2002**

Finite Automata

Regular expressions and finite automata have traditionally been used as syntactic representations of the regular languages over an alphabet Σ. The relationship between these two formalisms forms the basis of a well-developed classical theory. Classical developments range from the more combinatorial [9, 5, 7] to the more algebraic [11, 4, 1, 3, 10].

In this lecture we define the notion of an automaton over an arbitrary Kleene algebra. In subsequent sections, we will use this formalism to derive the classical results of the theory of finite automata (equivalence with regular expressions, determinization via the subset construction, elimination of ε-transitions, and state minimization) as consequences of the axioms of Kleene algebra.

Although we consider regular expressions and automata as syntactic objects, as a matter of convenience we will be reasoning modulo the axioms of Kleene algebra. Officially, regular expressions will denote elements of $\mathcal{F}_\Sigma$, the free Kleene algebra over Σ. The Kleene algebra $\mathcal{F}_\Sigma$ is constructed by taking the quotient of the regular expressions modulo the congruence generated by the axioms of Kleene algebra. The associated canonical map assigns to each regular expression its equivalence class in $\mathcal{F}_\Sigma$. Since we will be interpreting expressions only over Kleene algebras, and all interpretations factor through $\mathcal{F}_\Sigma$ via the canonical map, this usage is without loss of generality.

We recall the following basic theorems of Kleene algebra that were proved in Exercise **??** of Homework **??**, of which we will make extensive use:

$$\begin{aligned} xy = xz &\rightarrow x^*y = yz^* && (1)\\ (xy)^*x &= x(yx)^* && (2)\\ (x+y)^* &= x^*(yx^*)^*. && (3)\end{aligned}$$

These are called the *bisimulation rule*, the *sliding rule*, and the *denesting rule*, respectively.

Algebraic Definition of Finite Automata

Definition 0.1 A *finite automaton* over K is a triple $\mathcal{A} = (u, A, v)$, where $u, v \in \{0,1\}^n$ and $A \in \mathrm{Mat}(n, K)$ for some n.

The *states* are the row and column indices. The vector u determines the *start states* and the vector v determines the *final states*; a *start state* is an index i for which $u(i) = 1$ and a *final state* is one for which $v(i) = 1$. The $n \times n$ matrix A is called the *transition matrix*.

The *language accepted by* $\mathcal{A}$ is the element $u^T A^* v \in K$. □

For automata over $\mathcal{F}_\Sigma$, the free Kleene algebra on free generators Σ, this definition is essentially equivalent to the classical combinatorial definition of an automaton over the alphabet Σ as found in [9, 5]. A similar definition can be found in [2].

Example 0.2 Consider the two-state automaton in the sense of [9, 5] with states $\{p, q\}$, start state p, final state q, and transitions

$$p \xrightarrow{a} p \qquad q \xrightarrow{a} q \qquad p \xrightarrow{b} q \qquad q \xrightarrow{b} q.$$

Classically, this automaton accepts the set of strings over $\Sigma = \{a, b\}$ containing at least one occurrence of b. In our formalism, this automaton is specified by the triple

$$\left(\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} a & b \\ 0 & a+b \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right).$$

Modulo the axioms of Kleene algebra, we have

$$\begin{aligned} & \begin{bmatrix} 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} a & b \\ 0 & a+b \end{bmatrix}^* \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ = \; & \begin{bmatrix} 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} a^* & a^*b(a+b)^* \\ 0 & (a+b)^* \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ = \; & a^*b(a+b)^*. \end{aligned} \tag{4}$$

The language in Reg_Σ accepted by this automaton is the image under R_Σ of the expression (4). □

Definition 0.3 Let $\mathcal{A} = (u, A, v)$ be an automaton over $\mathcal{F}_\Sigma$, the free Kleene algebra on free generators Σ. The automaton $\mathcal{A}$ is said to be *simple* if A can be expressed as a sum

$$A \;=\; J + \sum_{a \in \Sigma} a \cdot A_a \tag{5}$$

where J and the A_a are 0-1 matrices. In addition, $\mathcal{A}$ is said to be *ε-free* if J is the zero matrix. Finally, $\mathcal{A}$ is said to be *deterministic* if it is simple and ε-free, and u and all rows of A_a have exactly one 1. □

In Definition 0.3, ε refers to the null string. The matrix A_a in (5) corresponds to the adjacency matrix of the graph consisting of edges labeled a in the combinatorial model of automata [5, 9] or the image of a under a linear representation map in the algebraic approach of [11, 1]. An automaton is deterministic according to this definition iff it is deterministic in the sense of [9, 5].

The automaton of Example 0.2 is simple, ε-free, and deterministic.

Completeness

In this section we prove the completeness of the axioms of Kleene algebra for the algebra of regular events. Another way of stating this is that Reg_Σ is isomorphic to $\mathcal{F}_\Sigma$, the free Kleene algebra on free generators Σ, and the standard interpretation $R_\Sigma : \mathcal{F}_\Sigma \to \mathsf{Reg}_\Sigma$ collapses to an isomorphism of Kleene algebras.

The first lemma asserts that Kleene's representation theorem [6, 1, 3, 10] is a theorem of Kleene algebra.

Lemma 0.4 *For every regular expression α over Σ (or more accurately, its image in $\mathcal{F}_\Sigma$ under the canonical map), there is a simple automaton (u, A, v) over $\mathcal{F}_\Sigma$ such that*

$$\alpha \quad = \quad u^T A^* v.$$

Proof. The proof is by induction on the structure of the regular expression. We essentially implement the combinatorial constructions as found for example in [5, 9]. The ideas behind this construction are well known and can be found for example in [2].

For $a \in \Sigma$, the automaton

$$\left(\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 & a \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right)$$

suffices, since

$$\begin{aligned} & \begin{bmatrix} 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & a \\ 0 & 0 \end{bmatrix}^* \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ = \;& \begin{bmatrix} 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ = \;& a. \end{aligned}$$

For the expression $\alpha + \beta$, let $\mathcal{A} = (u, A, v)$ and $\mathcal{B} = (s, B, t)$ be automata such that

$$\alpha = u^T A^* v \qquad \beta = s^T B^* t.$$

Consider the automaton with transition matrix

$$\left[\begin{array}{c|c} A & 0 \\ \hline 0 & B \end{array}\right]$$

and start and final state vectors

$$\left[\begin{array}{c} u \\ \hline s \end{array}\right] \quad \text{and} \quad \left[\begin{array}{c} v \\ \hline t \end{array}\right],$$

respectively. This construction corresponds to the combinatorial construction of forming the disjoint union of the two sets of states, taking the start states to be the union of the start states of $\mathcal{A}$ and $\mathcal{B}$, and the final states to be the union of the final states of $\mathcal{A}$ and $\mathcal{B}$. Then

$$\left[\begin{array}{c|c} A & 0 \\ \hline 0 & B \end{array}\right]^* = \left[\begin{array}{c|c} A^* & 0 \\ \hline 0 & B^* \end{array}\right],$$

and

$$\begin{aligned} & \left[\begin{array}{c|c} u^T & s^T \end{array}\right] \cdot \left[\begin{array}{c|c} A^* & 0 \\ \hline 0 & B^* \end{array}\right] \cdot \left[\begin{array}{c} v \\ \hline t \end{array}\right] \\ = \; & u^T A^* v + s^T B^* t \\ = \; & \alpha + \beta. \end{aligned}$$

For the expression $\alpha\beta$, let $\mathcal{A} = (u, A, v)$ and $\mathcal{B} = (s, B, t)$ be automata such that

$$\alpha = u^T A^* v \qquad \beta = s^T B^* t.$$

Consider the automaton with transition matrix

$$\left[\begin{array}{c|c} A & vs^T \\ \hline 0 & B \end{array}\right]$$

and start and final state vectors

$$\left[\begin{array}{c} u \\ \hline 0 \end{array}\right] \quad \text{and} \quad \left[\begin{array}{c} 0 \\ \hline t \end{array}\right],$$

respectively. This construction corresponds to the combinatorial construction of forming the disjoint union of the two sets of states, taking the start states to be the start states of $\mathcal{A}$, the final states to be the final states of $\mathcal{B}$, and connecting the final states of $\mathcal{A}$ with the start states of $\mathcal{B}$ by ε-transitions (this is the purpose of the vs^T in the upper right corner of the matrix). Then

$$\left[\begin{array}{c|c} A & vs^T \\ \hline 0 & B \end{array}\right]^* = \left[\begin{array}{c|c} A^* & A^* vs^T B^* \\ \hline 0 & B^* \end{array}\right],$$

and

$$\left[\begin{array}{c|c} u^T & 0 \end{array}\right] \cdot \left[\begin{array}{c|c} A^* & A^* v s^T B^* \\ \hline 0 & B^* \end{array}\right] \cdot \left[\begin{array}{c} 0 \\ \hline t \end{array}\right]$$

$$= u^T A^* v s^T B^* t$$

$$= \alpha\beta.$$

For the expression α^*, let $\mathcal{A} = (u, A, v)$ be an automaton such that $\alpha = u^T A^* v$. We first produce an automaton equivalent to the expression $\alpha\alpha^*$. Consider the automaton

$$(u,\ A + vu^T,\ v).$$

This construction corresponds to the combinatorial construction of adding ε-transitions from the final states of $\mathcal{A}$ back to the start states. Using (3) and (2),

$$\begin{aligned} u^T(A + vu^T)^* v &= u^T A^*(vu^T A^*)^* v \\ &= u^T A^* v(u^T A^* v)^* \\ &= \alpha\alpha^*. \end{aligned}$$

Once we have an automaton for $\alpha\alpha^*$, we can get an automaton for $\alpha^* = 1 + \alpha\alpha^*$ by the construction for $+$ given above, using a trivial one-state automaton for 1. $\square$

Now we get rid of ε-transitions. This construction is also folklore and can be found for example in [8, 10]. This construction models algebraically the combinatorial idea of computing the ε-closure of a state; see [5, 9].

Lemma 0.5 *For every simple automaton (u, A, v) over $\mathcal{F}_\Sigma$, there is a simple ε-free automaton (s, B, t) such that*

$$u^T A^* v \;=\; s^T B^* t.$$

Proof. By Definition 0.3, the matrix A can be written as a sum $A = J + A'$ where J is a 0-1 matrix and A' is ε-free. Then

$$\begin{aligned} u^T A^* v &= u^T (A' + J)^* v \\ &= u^T J^* (A' J^*)^* v \end{aligned}$$

by (3), so we can take

$$\begin{aligned} s^T &= u^T J^* \\ B &= A' J^* \\ t &= v. \end{aligned}$$

Note that J^* is 0-1 and therefore B is ε-free. $\square$

The next step in the proof will be to give algebraic analogs of the determinization of finite automata via the subset construction and the minimization of deterministic automata via the collapsing of equivalent states under a Myhill-Nerode equivalence relation. We will do this next time.

References

[1] Jean Berstel and Christophe Reutenauer. *Rational Series and Their Languages.* Springer-Verlag, Berlin, 1984.

[2] John Horton Conway. *Regular Algebra and Finite Machines.* Chapman and Hall, London, 1971.

[3] S. Eilenberg. *Automata, Languages, and Machines*, volume A. Academic Press, New York, 1974.

[4] F. Gécseg and I. Peák. *Algebraic Theory of Automata.* Akadémiai Kiadó, Budapest, 1972.

[5] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, 1979.

[6] Stephen C. Kleene. Representation of events in nerve nets and finite automata. In C. E. Shannon and J. McCarthy, editors, *Automata Studies*, pages 3–41. Princeton University Press, Princeton, N.J., 1956.

[7] Dexter Kozen. *Automata and Computability.* Springer-Verlag, New York, 1997.

[8] Werner Kuich and Arto Salomaa. *Semirings, Automata, and Languages.* Springer-Verlag, Berlin, 1986.

[9] Harry R. Lewis and Christos H. Papadimitriou. *Elements of the Theory of Computation.* Prentice Hall, 1981.

[10] Jacques Sakarovitch. Kleene's theorem revisited: A formal path from Kleene to Chomsky. In A. Kelemenova and J. Keleman, editors, *Trends, Techniques, and Problems in Theoretical Computer Science*, volume 281 of *Lecture Notes in Computer Science*, pages 39–50, New York, 1987. Springer-Verlag.

[11] Arto Salomaa and Matti Soittola. *Automata Theoretic Aspects of Formal Power Series.* Springer-Verlag, New York, 1978.