Operator Automata Theory

1 Introduction

The goal of operator automata theory is to view automata is to view automata as homomorphisms between linear spaces. In this sketch we develop the basic theory and preliminary results.

2 Preliminaries

2.1 Notation

(descriptive text here)

2.2 Algebra

Definition 1 (Monoid)**.** *A monoid $(M, \cdot, \mathbf{1})$ consists of a set M for which:*

- *Monoid multiplication $\cdot\colon M \times M \to M$ is associative.*
- *The element $\mathbf{1} \in M$ is the unique identity of multiplication.*

Definition 2 (Semiring)**.** *A semiring $(R, +, \cdot, \mathbf{0}, \mathbf{1})$ consists of a set R for which:*

- *Semiring multiplication $\cdot\colon R \times R \to R$ distributes over semiring addition $+\colon R \times R \to R$.*
- *$(R, +, \mathbf{0})$ is a commutative monoid.*
- *$(R, \cdot, \mathbf{1})$ is a monoid.*
- *Multiplication by zero $\mathbf{0}$ annihilates R.*

Definition 3 (Semimodule)**.** *For a semiring R, a R-semimodule $(M, +, \times, \mathbf{0})$ is such that:*

- *$(M, +, \mathbf{0})$ is a commutative group.*

- *Scalar multiplication* $\cdot\colon R \times M \to M$ *is a semiring action on* M*.*

Definition 4 (Group)**.** *A group* $(G, \cdot, \mathbf{1})$ *consists of a set* G *such that:*

- *Group multiplication* $\cdot\colon G \times G \to G$ *is associative.*
- $\mathbf{1}$ *is the identity under multiplication.*
- *If* $a \in G$*, then it has an inverse* $a^{-1} \in G$*.*

Definition 5 (Field)**.** *A field* $(K, +, \cdot, \mathbf{0}, \mathbf{1})$ *is a set* K *such that:*

- *Field multiplication* $\cdot\colon K \times K \to K$ *distributes over field addition* $+\colon K \times K \to K$*.*
- $(K, +, \mathbf{0})$ *is a commutative group.*
- $(K, \cdot, \mathbf{1})$ *is a commutative group.*
- *Multiplication by zero* $\mathbf{0}$ *annihilates* K*.*

Definition 6 (Linear Space)**.** *For a field* K*, a* K*-linear space* $(V, +, \cdot, \mathbf{0})$ *over a field* K *is a set such that:*

- $(V, +, \mathbf{0})$ *is a commutative group.*
- *Scalar multiplication* $\cdot\colon \mathbb{K} \times V \to V$ *is a field action on* V*.*

Definition 7 (Linear Operator)**.** *For* K*-linear spaces* V *and* W*, a linear operator* $T\colon V \to W$ *is a homomorphism.*

Definition 8 (Non-Deterministic Finite Automata)**.** *A non-deterministic finite automata (NFA)* $(\Sigma, Q, \Delta, S, F)$ *is a finite state machine that accepts or rejects strings. A NFA reads from a tape of finite tape of cells in which each cell has a letter from the finite alphabet* Σ*. At any point, the NFA is in some state* Q*, and upon reading the current tape cell's contents the transition function* $\Delta\colon \Sigma \times Q \to 2^Q$ *will transition the NFA non-deterministically to any one of the possible next states in addition to advancing to read the next cell. The set of start states* $S \subseteq Q$ *is the set of states that the NFA may begin at, while the set of final states* $F \subseteq Q$ *is the state of states that the NFA must be in in order to accept a string after reading the last letter on the tape.*

Definition 9 (Norm)**.** *A normed space* $(V, \|\cdot\|)$ *is a* K*-linear space* V *with a norm function* $\|\cdot\| : V \to \mathbb{R}_{\geq 0}$ *such that:*

- *For all* $x, y \in V$*,* $\|x + y\| \leq \|x\| + \|y\|$*.*
- *For all* $a \in K$ *and* $x \in V$*,* $\|ax\| = |a| \cdot \|v\|$*.*
- *For* $\mathbf{0} \in V$*,* $\|\mathbf{0}\| = 0$*.*

3 Automata as Operators

We leverage some of the algebraic sections previously discussed to construct representations relevant to us.

3.1 Algebraic Structures of Automata

For a NFA A, with $A = (\Sigma, Q, \Delta, S, F)$, the goal is to view A acting as a linear operator between spaces. In particular, the approach we take is to see Δ as a linear operator between spaces: this is perhaps the most obvious approach, because to begin with, Δ is already the transition function.

But the questions are then: what are the appropriate linear spaces, and what are the actions of the linear operator? One initial thought is that the transition function can, in some way, be seen as a directed acyclic graph on the states Q, where each edge is weighted by the letters in the alphabet Σ that induce the transition. The representation as an adjacency matrix does appear in literature [1]. Roughly, if M is the transition matrix, then $M_{i,j} \in 2^{\Sigma}$ denotes the states that will transition state q_i to q_j.

While such a matrix representation is useful, it is not immediately obvious how such a matrix does indeed correspond to a linear operator, especially on what linear spaces. One possible interpretation is to see the linear spaces as $|Q|$-dimensional, where each dimension of the space corresponds to one member of Q. The objects of the space is then sets of strings over Σ.

Definition 10 (String Space)**.** *The string space of* Σ *is a semiring* $\left(2^{\Sigma^\star}, \cup, \cdot, \mathbf{0}, \mathbf{1}\right)$ *such that:*

- *The semiring addition is the set union* $\cup \colon 2^{\Sigma^\star} \times 2^{\Sigma^\star} \to 2^{\Sigma^\star}$.
- *The semiring multiplication is the string concatenation* $\cdot \colon 2^{\Sigma^\star} \times 2^{\Sigma^\star} \to 2^{\Sigma^\star}$ *such that:*

$$A \cdot B = \{a \cdot b \ : a \in A, b \in B\}$$

Here the string space is the power set of all strings generated by Σ through monoid multiplication (string concatenation). Defining the string space like this allows us to equip it with a semiring structure by viewing addition as set union. For convenience, we may write R instead of $2^{\Sigma^\star}$ to denote the set corresponding to the string space.

Observe that R has the structure of a 1-dimensional linear space. The big difference, however, is that the scalar elements are elements of a semiring rather than a field. Nevertheless, R is still closed under linear operations, and is therefore a linear space.

Theorem 1. *A string space* R *is a linear space.*

Proof. **(descriptive text here)** $\square$

The natural extension of a 1-dimensional linear space is a n-dimensional linear space.

Definition 11 (n-String Space)**.** *For a string space R and $n \in \mathbb{Z}_+$, the n-dimensional string space R^n is then the free semimodule isomorphic to n copies of R.*

A natural representation of R^n is as a n-dimensional vector, and in this case we prefer row vectors to column vectors. We abuse notation to identify elements of R^n with their row vector representation. Furthermore, we demonstrate that this is indeed still a linear space.

Theorem 2. *A string space R^n is a linear space.*

Proof. **(descriptive text)** $\square$

In particular, it would be nice to have a linear operator between string spaces.

Definition 12 (Linear String Space Operator)**.** *A linear string space operator is a linear operator $A\colon R^n \to R^m$.*

In particular, we are interested in a matrix representation.

Definition 13 (Matrix Representation of Linear String Space Operator)**.** *The matrix representation of a linear string space operator $A\colon R^n \to R^m$ is a matrix $A \in M_{n\times m}(R)$ that acts on row vectors of R^n by right multiplication.*

Here each entry of the matrix denotes the sets strings that are concatenated during transition. As with the n-dimensional string space R^n, we abuse notation for A to stand in for both the linear operator and its matrix representation. We now show that this is indeed a linear operator.

Theorem 3. *The matrix representation of $A\colon R^n \to R^m$ is a linear operator.*

Proof. **(descriptive text)** $\square$

Observe that the elements for the matrix of A are drawn from R (which is just $2^{\Sigma^\star}$) rather than 2^Σ, which is what we would expect for an NFA. In other words, transitions in A are given by sets of potentially long strings, rather than just single alphabets. The definition provided here is intended to be slightly more general, with the NFA case of single-letter transitions being a special case. Nevertheless, given a NFA, we are now ready to describe a particular matrix representation as a linear operator between n-string spaces.

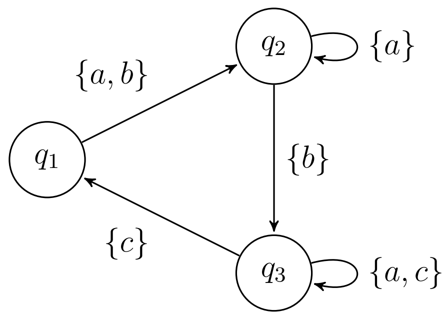
Definition 14 (Matrix Representation of Δ)**.** *For a NFA $(\Sigma, Q, \Delta, S, F)$ with string space R generated by Σ and $n = |Q|$, the matrix representation of Δ as a linear operator is a matrix $A \in M_{n\times n}(R)$ where:*

$$A_{i,j} = \{a \,:\, ((q_i, a), B) \in \Delta, q_j \in B\}$$

Of course, this is a linear string space operator simply because every entry of the transition matrix will be a set of singleton strings.

3.2 Examples

Example 1. *Consider* $\Sigma = \{a, b, c\}$ *and the NFA below:*



The most important distinction between this and a typical NFA is that we implicitly assume every state to be both starting and accepting. In other words, accepting strings is very permissive, which simplifies things for now. However, to embed this into our model there are several steps.

First, we have the string space $\left(2^{\Sigma^\star}, \bigcup, \cdot, \mathbf{0}, \mathbf{1}\right)$.

Next, for the matrix A *that we will construct, take* $A_{i,j}$ *to denote the transition from state* q_i *to state* q_j*. The matrix is then:*

$$A = \begin{bmatrix} \mathbf{0} & \{a, b\} & \mathbf{0} \\ \mathbf{0} & \{a\} & \{b\} \\ \{c\} & \mathbf{0} & \{a, c\} \end{bmatrix}$$

In order to perform string concatenation towards the right, transition matrices act by right-matrix multiplication. That is, if $v \in R^3$ *is the initial* n*-dimensional string space, then the subsequent string space is* vA*.*

To briefly demonstrate, two transitions of the matrix A *appears as follows:*

$$A^2 = \begin{bmatrix} \mathbf{0} & \{a, b\} & \mathbf{0} \\ \mathbf{0} & \{a\} & \{b\} \\ \{c\} & \mathbf{0} & \{a, c\} \end{bmatrix} \begin{bmatrix} \mathbf{0} & \{a, b\} & \mathbf{0} \\ \mathbf{0} & \{a\} & \{b\} \\ \{c\} & \mathbf{0} & \{a, c\} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \{aa, ba\} & \{ab, bb\} \\ \{bc\} & \{aa\} & \{ab, ba, bc\} \\ \{ac, cc\} & \{ca, cb\} & \{aa, ac, ca, cc\} \end{bmatrix}$$

In general, from graph theory, A^k *denotes the* k*th consecutive transition using* A*, and each entry* $A^k_{i,j}$ *is the set of strings that will get from state* q_i *to* q_j *in* k *steps.*

References

[1] John E Savage. *Models of computation.* Vol. 136. Addison-Wesley Reading, MA, 1998.