AUTOMATA THEORY AND GENERATING FUNCTIONS

ATTICUS KUHN

Date: December 6, 2021.

CONTENTS

1. INTRODUCTION

There are many interesting word puzzles in newspapers, such as find the number of strings of 1 and 0 with an even number of 0s, and this hints at a much deeper area of study, known as automata theory. In this paper, we will be introduced to generating functions, automata theory, and regular languages. We will discover several interesting properties and theorems, and we will use this to systematically answer such questions as was just posed.

2. AUTOMATA

First, Let us define an automata. There are 5 major components to a regular automata, which are

(1) Q is a finite set of states,
(2) $q_0 \in Q$ is the start state,
(3) $A \subset Q$is a distinguished set of accepting states,
(4) Σ is a finite input alphabet,
(5) δ is a function from $Q \times \Sigma$ into Q, called the transition function of M.

The way an automata works is it starts at the initial state q_0. It is then fed the string 1 character at a time. The transition function δ tells it how to go from one state to the next. If it ends at an accepting state, the string is accepted, otherwise it is rejected. Let us start with a simple example In 1, it will accept the string 1100 because it will end at q_1 which is an accepting state, but it will

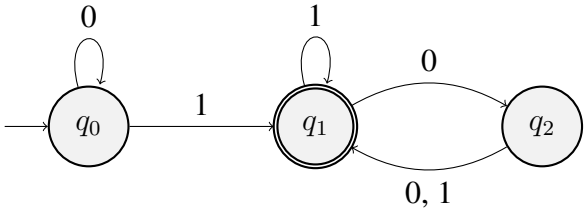

FIGURE 1. An example automata

not accept the string 0010 because then it would end at state q_2 and would be rejected.

3. REGULAR LANGUAGES

Among languages, that is sets of finite alphabets, a regular language is any language that can be recognized by a regular expression, or said another way

Definition 1. *A regular language is A regular expression over and alphabet Σ is built from Σ and the symbols "(", ")", "ϵ", "+", and "*", according to the following recursive definition:*

- *the empty string ϵ and each member of Σ is a regular expression;*
- *if α and β are regular expressions then so is the union of α and β ($\alpha\beta$);*
- *if α and β are regular expressions then so is α concatenated to β (α + β);*
- *if α is a regular expression then so is $\alpha*$ (the Kleene star)*

The way to relate Automata to regular languages is with Kleene's Theorem, which says that

Theorem 2. *(Kleene's Theorem) Any language that can be defined by a regular expression or finite state machine is a regular language. Every regular expression has an equivalent finite state machine and vice versa.*

One way of expressing a regular expression is with *Extended Backus–Naur form* or EBNF. It has several symbols, such as * which means 0 or more times and | means a union. For example $A * (B|C)$ would mean any number of A followed by either a B or a C.

4. Generating function

Generating functions serve as a useful way of turning a sequence into a function.

Definition 3. *For a sequence a_n, its **Generating Function** is*

$$A(z) = \sum_{k=0}^{\infty} a_k z^k$$

If we are counting objects, then a_n is usually the number of objects of size n. Even though this is a function, we don't plug values into it, it is a way of packaging up a sequence.

Example 4. *A simple example of a generating function for a regular language would be the language that accepts all strings. If $|A| = m$, then*

$$A(z) = \sum_{k=0}^{\infty} m^k z^k = \frac{1}{1 - mz}$$

4.1. Examples.

Let us look at how to find the generating function of the regular expression $1 * 00 * 1(0|1)*$ it has the automata diagram

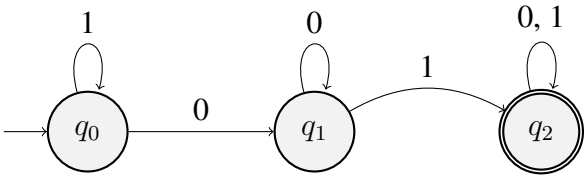

FIGURE 2. An example automata

We can use 4 to help us solve this. From q_0, $\frac{1}{1-z}$ is the generating function from q_0 to q_0, and z is the generating function from q_0 to q_1. We can use similar logic for q_1. For q_2, both 0 and 1 go back to q_2, so it is $\frac{1}{1-2x}$. We combine these to get

$$F_L(z) = (\frac{1}{1-z}z)(\frac{1}{1-z}z)(\frac{1}{1-2z})$$

We can use partial fractions decomposition to get that

$$F_L(z) = \frac{-1}{(1-z)^2} + \frac{1}{1-2z} = -\sum_{n=1}^{\infty} nx^{n-1} + \sum_{n=0}^{\infty} 2^n x^n = \sum_{n=0}^{\infty} (2^n - n - 1)x^n$$

Meaning that $f_L = 2^n - n - 1$

5. Linear Recurrence

We can analyze automata if we look at them in terms of linear recurrences. Let us analyze the regular expression $1 * 00 * 1(01) * 1(0|1)*$ This gives us the automata let $L_a(z)$ represent

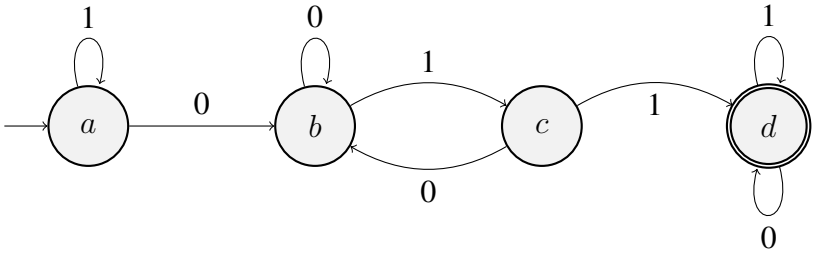

FIGURE 3. An example automata

the generating function for counting the number of ways to get to d, the accepting state, from a. For example $L_a(z) = zL_b(z) + zL_a(z)$ because from a we can either go to b in 1 step or a in 1 step. We can write out the remaining equations as follows. $L_a(z) = zL_b(z) + zL_a(z), L_b(z) = zL_b(z) + zL_c(z), L_c(z) = zL_b(z) + zL_d(z), L_d(z) = zL_d(z) + zL_d(z) + 1$ Note that we add 1 to $L_d(z)$ because 1 represents the empty move and at d we can just do nothing. In terms of matrices, we can write it as

$$\begin{bmatrix} L_a \\ L_b \\ L_c \\ L_d \end{bmatrix} = \begin{bmatrix} z & z & 0 & 0 \\ 0 & z & z & 0 \\ 0 & z & 0 & 0 \\ 0 & 0 & 0 & 2z \end{bmatrix} \begin{bmatrix} L_a \\ L_b \\ L_c \\ K_d \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

We can solve this in using Gaussian elimination or any method to solve it.

In other words:

$$\mathrm{L}_a(z) = \frac{z^3}{(1-z)(1-2z)(1-z-z^2)} L_b(z) = \frac{z^2}{(1-2z)(1-z-z^2)} L_c(z) = \frac{z(1-z)}{(1-z-z^2)(1-2z)} L_d(z) = \frac{1}{1-2z}$$

We are only interested in $L_a(z)$, of course, because a is the starting state. To find the coefficients. Let us use paritial fraction decomposition to get

$$L_a = \frac{1}{1-z} + \frac{1}{1-2z} - \frac{z+2}{1-z-z^2}$$

We then use power series to see that

$$L_a = \sum_{k=0}^{\infty} z^k + \sum_{k=0}^{\infty} 2^k z^k + \sum_{k=0}^{\infty} F_{k+3} z^n$$

where F_n is the nth Fibonacci number. We can then see that $l_a = 2^n + 1 - F_{n+3}$

6. Conclusion

This paper has covered an introduction to automata theory and generating functions, and the field goes much deeper. Automata theory is very connected to the field of computer science. For those who would like to study the automata theory further, I reccomend looking at it through the lens of compute science, which is very interesting.

References

[1] Ferran Alet Puig. Generating functions and searching automata.

[2] Benjamin Steinberg. Notes on generating functions in automata theory. 2009.

[3] Larry Joseph Stockmeyer. *The complexity of decision problems in automata theory and logic.* PhD thesis, Massachusetts Institute of Technology, 1974.

[4] Wikipedia contributors. Automata theory — Wikipedia, the free encyclopedia, 2021. [Online; accessed 7-November-2021].

[5] Wikipedia contributors. Generating function — Wikipedia, the free encyclopedia, 2021. [Online; accessed 7-November-2021].

[6] Wikipedia contributors. Regular language — Wikipedia, the free encyclopedia, 2021. [Online; accessed 7-November-2021].

[7] Carol Zander. Languages (finite state machines). 2012.

[]