CS 311: Computational Structures

James Hook

October 15, 2015

5 Regular Expressions and Pumping Lemma

5.1 Recall

- Regular Expressions (REGEXPs)
- Every REGEXP describes a Regular Language.
- Every Regular Language is described by a REGEXP. Started sketch.

5.2 Plan

- Discuss Problem Set 2
- The GNFA Construction
- Pumping Lemma for Regular Languages

5.3 Problem Set 2

Lower bound problem.

Shuffle.

5.4 Regular Expressions

Review strategy.

Look at definition of acceptance for GNFA.

Review key formula for ripping out a state.

5.5 GNFA Example in Detail

This note illustrates the application of the construction in Sipser's proof of Lemma 1.60 to the modulo 3 counter DFA.

In lecture we have shown the DFA that recognizes binary numbers modulo 3. $M = (\{0,1,2\}, \{0,1\}, \delta, 0, \{2\})$, where δ is given by the table:

$$\begin{aligned}
\delta(0,0) &= 0\\
\delta(0,1) &= 1\\
\delta(1,0) &= 2\\
\delta(1,1) &= 0\\
\delta(2,0) &= 1\\
\delta(2,1) &= 2
\end{aligned}$$

Convert this to a GNFA by adding states s and a, and labeling all transitions with regular expressions. The GNFA can be seen by this table, where each row is the "from" state and each column is the "to" state:

δ	0	1	2	a
s	ϵ	$\emptyset$	$\emptyset$	$\emptyset$
0	0	1	$\emptyset$	$\emptyset$
1	1	$\emptyset$	0	$\emptyset$
2	$\emptyset$	0	1	ϵ

To "rip" state 0 out of this machine, we calculate the following δ':

δ'	1	2	a
s	$(\delta(s,0))(\delta(0,0))^*(\delta(0,1)) \cup (\delta(s,1))$	$(\delta(s,0))(\delta(0,0))^*(\delta(0,2)) \cup (\delta(s,2))$	$(\delta(s,0))(\delta(0,0))^*(\delta(0,a)) \cup (\delta(s,a))$
1	$(\delta(1,0))(\delta(0,0))^*(\delta(0,1)) \cup (\delta(1,1))$	$(\delta(1,0))(\delta(0,0))^*(\delta(0,2)) \cup (\delta(1,2))$	$(\delta(1,0))(\delta(0,0))^*(\delta(0,a)) \cup (\delta(1,a))$
2	$(\delta(2,0))(\delta(0,0))^*(\delta(0,1)) \cup (\delta(2,1))$	$(\delta(2,0))(\delta(0,0))^*(\delta(0,2)) \cup (\delta(2,2))$	$(\delta(2,0))(\delta(0,0))^*(\delta(0,a)) \cup (\delta(2,a))$

δ'	1	2	a
s	$\epsilon 0^*1 \cup \emptyset$	$\epsilon 0^*\emptyset \cup \emptyset$	$\epsilon 0^*\emptyset \cup \emptyset$
1	$10^*1 \cup \emptyset$	$10^*\emptyset \cup 0$	$10^*\emptyset \cup \emptyset$
2	$\emptyset 0^*1 \cup 0$	$\emptyset 0^*\emptyset \cup 1$	$\emptyset 0^*\emptyset \cup \epsilon$

δ'	1	2	a
s	0^*1	$\emptyset$	$\emptyset$
1	10^*1	0	$\emptyset$
2	0	1	ϵ

Next, rip state 2 out:

δ''	1	a
s	$(\delta'(s,2))(\delta'(2,2))^*(\delta'(2,1)) \cup (\delta'(s,1))$	$(\delta'(s,2))(\delta'(2,2))^*(\delta'(2,a)) \cup (\delta'(s,a))$
1	$(\delta'(1,2))(\delta'(2,2))^*(\delta'(2,1)) \cup (\delta'(1,1))$	$(\delta'(1,2))(\delta'(2,2))^*(\delta'(2,a)) \cup (\delta'(1,a))$

δ''	1	a
s	$\emptyset 1^*0 \cup 0^*1$	$\emptyset 1^*\epsilon \cup \emptyset$
1	$01^*0 \cup 10^*1$	$01^*\epsilon \cup \emptyset$

δ''	1	a
s	0^*1	$\emptyset$
1	$01^*0 \cup 10^*1$	01^*

Finally, we rip out state 1, leaving the single transition:

$$(\delta''(s,1))(\delta''(1,1))^*(\delta''(1,a)) \cup (\delta''(s,a))$$

Which becomes:

$$0^*1(01^*0 \cup 10^*1)^*01^* \cup \emptyset$$

Or simply:

$$0^*1(01^*0 \cup 10^*1)^*01^*$$

The result is a regular expression generating the set of binary numbers that are congruent to 2 modulo 3.

Exercise 5.1 *What if we ripped the states in a different order?*

5.6 Pumping Lemma

Ultimate goal of this section is to find a way to say that a language is not regular. We begin by identifying a property that all regular languages have. We will then be able to show that any language that does not have that property is not regular.

When are regular language infinite?

When they are infinite, how can be build “new strings from old”?

Pumping Lemma characterizes that for any regular language, if it contains a sufficiently long string, we can build an infinite number of strings in the language by repeating substrings.

The pumping lemma can be applied to show that some languages are not regular, because they fail to include