

Regular Languages

Dr. Mattox Beckman

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
DEPARTMENT OF COMPUTER SCIENCE



Objectives

You should be able to ...

- ▶ Use the syntax of regular expressions to model a given set of strings.
- ▶ Give examples of the limitations of regular expressions.

Motivation

- ▶ *Regular languages* were developed by Noam Chomsky in his quest to describe human languages.
- ▶ Computer scientists like them because they are able to describe “words” or “tokens” very easily.

Examples:

Integers a bunch of digits

Reals an integer, a dot, and an integer

Past Tense English Verbs a bunch of letters ending with “ed”

Proper Nouns a bunch of letters, the first of which must be capitalized

A Bunch of Digits?!

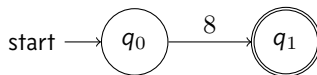
- ▶ We need something a bit more formal if we want to communicate properly.
- ▶ We will use a *pattern* (or a *regular expression*) to represent the kinds of words we want to describe.
- ▶ These expressions will correspond to NFAs.
- ▶ Kinds of patterns we will use:
 - ▶ Single letters
 - ▶ Repetition
 - ▶ Grouping
 - ▶ Choices

Single Letters

- ▶ To match a single character, just write the character.
- ▶ To match the letter "a" ...
 - ▶ Regular expression: a
 - ▶ State machine:

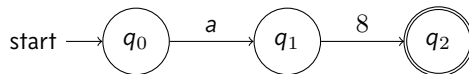


- ▶ To match the character "8" ...
 - ▶ Regular expression: 8
 - ▶ State machine:

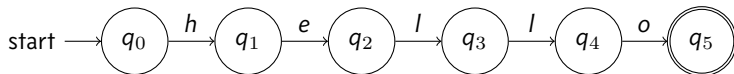


Juxtaposition

- ▶ To match longer things, just put two regular expressions together.
- ▶ To match the character "a" followed by the character "8" ...
 - ▶ Regular expression: a8
 - ▶ State machine:

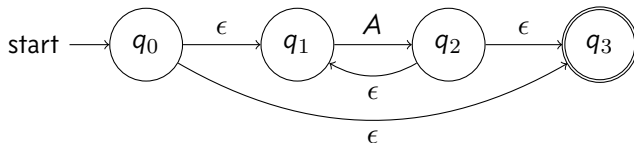


- ▶ To match the string "hello" ...
 - ▶ Regular expression: hello
 - ▶ State machine:

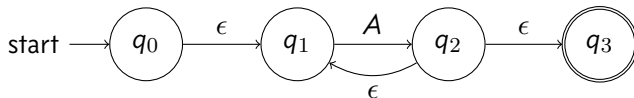


Repetition

- ▶ Zero or more copies of A , add $*$
 - ▶ Regular expression A^*
 - ▶ State machine:

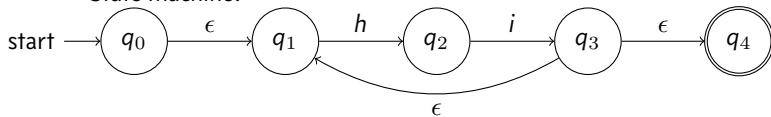


- ▶ One or more copies of A , add $+$
 - ▶ Regular expression A^+
 - ▶ State machine:



Grouping

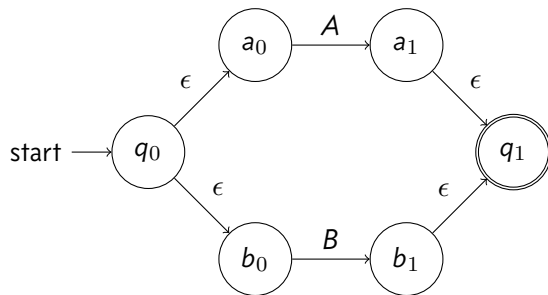
- ▶ To group things together, use parenthesis.
- ▶ To match one or more copies of the word "hi" ...
 - ▶ Regular expression: $(hi)^+$
 - ▶ State machine:



- ▶ We use Thompson's construction to build the state machine. The extra ϵ transitions are important!

Choice

- ▶ To make a choice, use the vertical bar (also called “pipe”).
- ▶ To match A or B ...
 - ▶ Regular expression: $A|B$
 - ▶ State machine:



Examples

Expression	(Some) Matches	(Some) Rejects
ab^*a	aa, aba, abbba	ba, aaba, abaa
$(0 1)^*$	any binary number, ϵ	
$(0 1)^+$	any binary number	empty string
$(0 1)^*0$	even binary numbers	
$(aa)^*a$	odd number of as	
$(aa)^*a(aa)^*$	odd number of as	
$(aa bb)^*((ab ba)(aa bb)^*(ab ba)(aa bb)^*)^*$	even number of as and b	

Some Notational Shortcuts

- ▶ A range of characters: `[Xa-z]` matches X and between a and z (inclusively).
- ▶ Any character at all: `.`
- ▶ Escape: `\`

Expression

`[0-9]+`

`X.*Y`

`[0-9]*\.[0-9]*`

(Some) Matches

integers

anything at all between an X and a Y

floating point numbers (positive, without exponents)

Things to Know ...

- ▶ They are *greedy*.
`X.*Y` will match `XabaaYaababY` entirely, not just `XabaaY`.
- ▶ They *cannot count* very well.
 - ▶ They can only count as high as you have states in the machine.
 - ▶ This regular expression matches some primes:
`aa | aaa | aaaaa | aaaaaaa`
 - ▶ You cannot match an infinite number of primes.
 - ▶ You cannot match “nested comments.” (`*.**`)