BU CS 332 – Theory of Computation

Lecture 5:

- More on pumping
- Regular expressions
- Regular expressions = regular languages

Reading:

Sipser Ch 1.3

Mark Bun

February 5, 2020

More on Pumping

Pumping Lemma (Formal)

Let L be a regular language.

Then there exists a “pumping length” p such that

For every $w \in L$ where $|w| \geq p$,
w can be split into three parts $w = xyz$ where:

1. $|y| > 0$
2. $|xy| \leq p$
3. $xy^iz \in L$ for all $i \geq 0$

General Strategy for proving L is not regular

Proof by contradiction: assume L is regular.

Then there is a pumping length p.

1. Find $w \in L$ with $|w| \geq p$

2. Show that w cannot be pumped

3. Conclude L must not have been regular

Pumping down

Proof that actually works

Pump 0 times:

$xz = 0^m 0^{p+1-(m+k)} 1^p$

$= 0^{p+1-k} 1^p$

$\notin L$ ~~$(p+1-k \le p)$~~

Claim: $L = \{0^i 1^j \mid i > j \geq 0\}$ is not regular

Proof: Assume L is regular with pumping length p

1. Find $w \in L$ with $|w| \geq p$ $\quad 0^{p+1} 1^p$

2. Show that w cannot be pumped

   Formally If $w = xyz$ with $|xy| \leq p$, then... $|y| > 0$

$$\overbrace{0\,0\,0\,0\,0\,0\,0\,0 \ldots 0\,0\,0}^{p+1}\,\overbrace{1\,1\,1\,1\,1\,1\,1\,1}^{p}$$

x y z

$y = 0^k \quad k > 0$

$x = 0^m \quad m \geq 0$

$k + m \leq p$

$z = 0^{p+1-(m+k)} 1^p$

$xyyz = \underbrace{0^m}_{x} \underbrace{0^k}_{y} \underbrace{0^k}_{y} \underbrace{0^{p+1-(m+k)} 1^p}_{z}$

$= 0^{p+1+k} 1^p \in L$

$xyyyz \quad xyyyyz$

Reusing a Proof

$ALL = \Sigma^*$

Pumping a language can be lots of work…

Let's try to reuse that work!

Not regular
$\{0^n1^n \mid n \geq 0\}$

How might we show that

$BALANCED$ = $\{w \mid w$ has an equal # of 0s and 1s$\}$

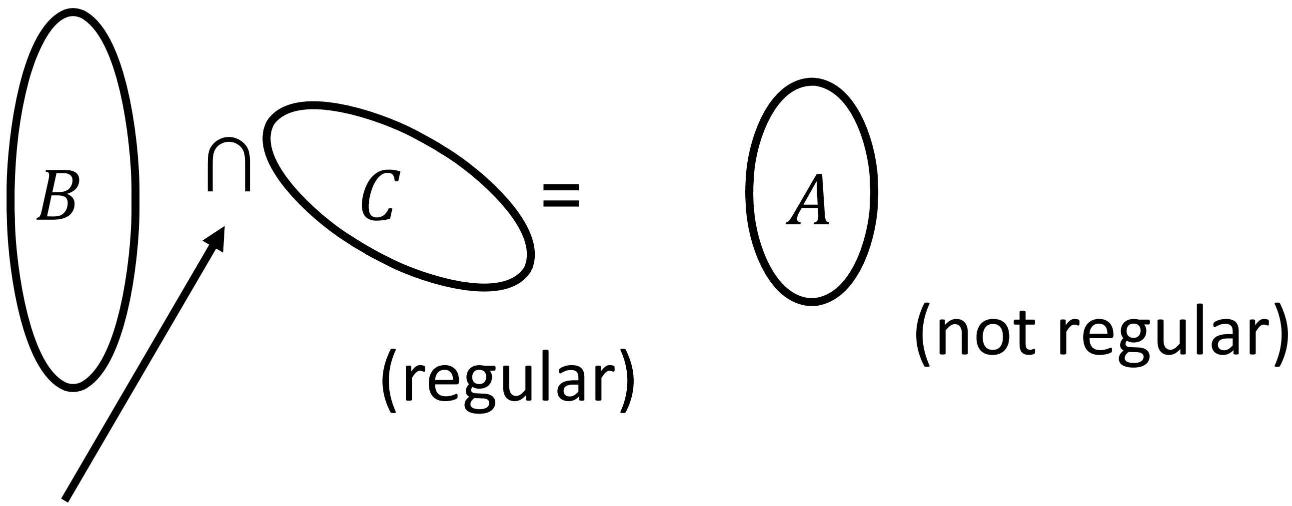
is not regular?

$$\underbrace{\{0^n1^n \mid n \geq 0\}}_{\text{Not regular}} = \underbrace{BALANCED}_{???} \cap \underbrace{\{w \mid \text{all 0s in } w \text{ appear before all 1s}\}}_{\text{regular}}$$

Assume for contradiction BALANCED is regular $\Rightarrow$ RHS regular (closure under $\cap$) $\times$
$\Rightarrow$ BALANCED not regular

Using Closure Properties

If A is not regular, we can show a related language B is not regular



any of $\{\circ, \cup, \cap\}$ or, for one language, $\{\neg, {}^{R}, *\}$

<u>By contradiction:</u> If B is regular, then $B \cap C$ $(= A)$ is regular.

But A is not regular so neither is B!

Example

Universe $= \Sigma^*$ $\quad C = A \cup B$

$A = C \setminus B, \; B = C \setminus A$

Prove $B = \{0^i 1^j | i \neq j\}$ is not regular

using nonregular language $A = \{0^n 1^n | n \geq 0\}$

and regular language

$C = \{w \mid$ all 0s in w appear before all 1s$\}$

$\bar{A} = \{w \in \{0,1\}^* \mid w = 0^i 1^j \; i \neq j$ or w mixes 0's and 1's$\}$

$\cancel{B = \bar{A} \cap C}$

$A = \bar{B} \cap C$

non-regular

B regular $\Rightarrow$ A regular. ✗

Regular Expressions

Regular Expressions

- A different way of describing regular languages
- A regular expression expresses a (possibly complex) language by combining simple languages using the regular operations

"Simple" languages: $\emptyset$, $\{\varepsilon\}$, $\{a\}$ for some $a \in \Sigma$

Regular operations:

Union: $A \cup B$

Concatenation: $A \circ B = \{ab \mid a \in A, b \in B\}$

Star: $A^* = \{ a_1 a_2 \ldots a_n \mid n \geq 0 \text{ and } a_i \in A\}$

$= \{\varepsilon\} \cup A \cup AA \cup AAA \cup \ldots$

Regular Expressions – Syntax

A regular expression R is defined recursively using the following rules:

1. ε, $\emptyset$, and a are regular expressions for every $a \in \Sigma$

"concatenate" "star"

2. If R_1 and R_2 are regular expressions, then so are
   $(R_1 \cup R_2)$, $(R_1 \circ R_2)$, and (R_1^*)

Examples: (over $\Sigma = \{a, b, c\}$)
$(a \circ b)$ $\quad$ $((((a \circ (b^*)) \circ c) \cup (((a^*) \circ b))^*))$ $\quad$ $(\emptyset^*)$

Regular Expressions – Semantics

$L(R)$ = the language a regular expression describes

$L(a^*) = (L(a))^* = (\{a\})^*$
$= \{\epsilon, a, aa, aaa, \ldots\}$
$= \{\underbrace{aaa..a}_{n} \mid n \geq 0\}$

1. $L(\emptyset) = \emptyset$
2. $L(\varepsilon) = \{\varepsilon\}$
3. $L(a) = \{a\}$ for every $a \in \Sigma$
4. $L((R_1 \cup R_2)) = L(R_1) \cup L(R_2)$
5. $L((R_1 \circ R_2)) = L(R_1) \circ L(R_2)$
6. $L\left((R_1^*)\right) = (L(R_1))^*$

$A^* = \{w_1 w_2 \ldots w_n \mid n \geq 0, w_i \in A \ \forall i\}$

Example: $L(((a^*) \circ (b^*))) =$  $\{a^i b^j \mid i, j \geq 0\}$

Simplifying Notation

- Omit $\circ$ symbol: $(ab) = (a \circ b)$ $R_1 R_2 = R_1 \circ R_2$

- Omit many parentheses, since union and concatenation are associative:

$$(a \cup b \cup c) = (a \cup (b \cup c)) = ((a \cup b) \cup c)$$

- Order of operations: Evaluate star, then concatenation, then union

$$ab^* \cup c = (a(b^*)) \cup c$$

$$((a^*) \circ (b^*)) = a^* b^*$$

Examples

$\Sigma_1^{*} \Rightarrow 0101$

Let $\Sigma = \{0, 1\}$

1. $\{w \mid w \text{ contains exactly one } 1\}$

   $0^* 1 0^*$

2. $\{w \mid w \text{ has length at least 3 and its third symbol is } 0\}$

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

3. $\{w \mid \text{every odd position of } w \text{ is } 1\}$ $\quad w_0 w_1 w_2 w_3 w_4 \ldots$

   $((0 \cup 1) 1)^* (\epsilon \cup 0 \cup 1)$ $\quad 0\ 1\ 0$

Syntactic Sugar

- For alphabet Σ, the regex Σ represents $L(\Sigma) = \Sigma$
- For regex R, the regex $R^+ = RR^*$

$R^* = R^+ \cup \varepsilon$

Not captured by regular expressions: Backreferences

\1 \2

Equivalence of Regular Expressions, NFAs, and DFAs

Regular Expressions Describe Regular Languages

Theorem: A language A is regular if and only if it is described by a regular expression

Theorem 1: Every regular expression has an equivalent NFA

Theorem 2: Every NFA has an equivalent regular expression

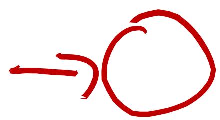
Regular expression -> NFA

Theorem 1: Every regex has an equivalent NFA

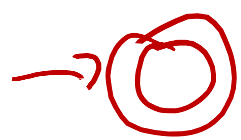
Proof: Induction on size of a regex

Base cases:

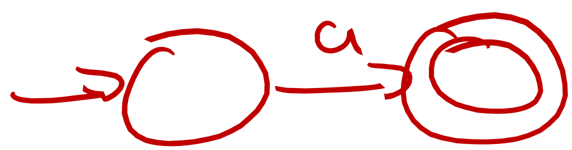
$R = \emptyset$



$R = \varepsilon$



$R = a$



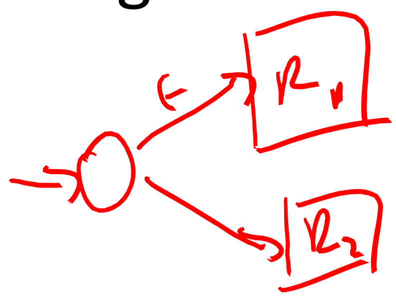
Regular expression -> NFA

Theorem 1: Every regex has an equivalent NFA

Proof: Induction on size of a regex

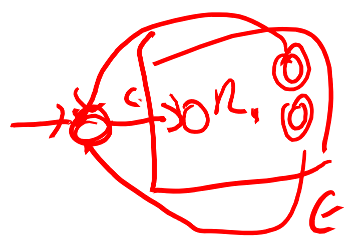
Inductive step:

$$R = (R_1 \cup R_2)$$



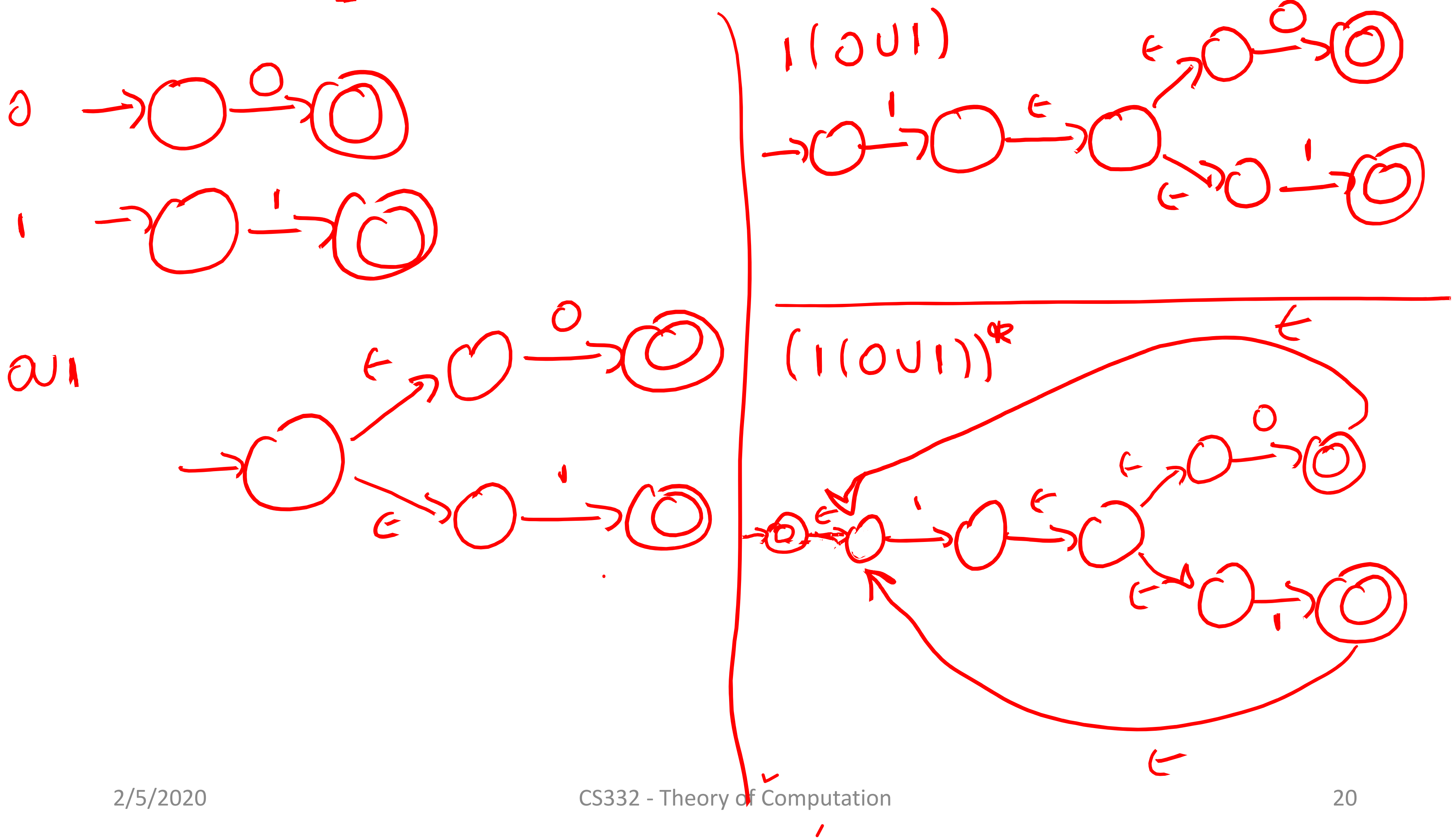
$$R = (R_1 R_2)$$

$$R = (R_1^*)$$



Example

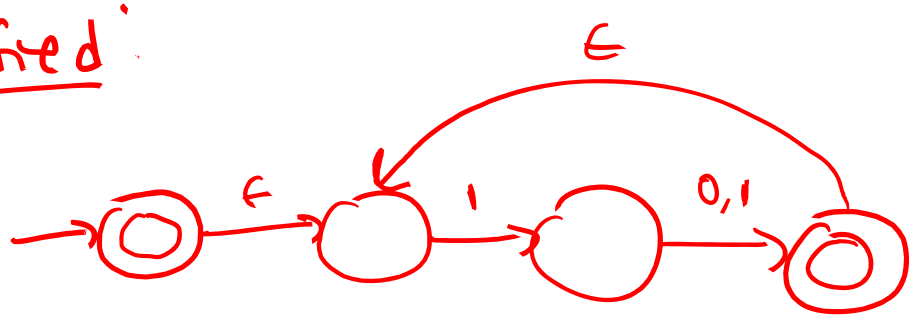
Convert $(1(0 \cup 1))^*$ to an NFA



Example

Simplified:

=



=