BU CS 332 – Theory of Computation

Lecture 5:

- Closure Pro
- Regular Expressions

Reading:

pser Ch 1.2-1.3

Mark Bun

February 8, 2021

Last Time

- NFAs vs. DFAs
  - Subset construction: NFA -> DFA

- Intro to closure anguages

Closur s

Operations on languages

Let $A, B \subseteq \Sigma^*$ be languages. Define

Regular Operations

- Union: $A \cup B$
- Concatenation: $A \circ B = \{ab \mid a \in A, b \in B\}$
- Star: * and $a_i \in A\}$

- Com

- Intersection: $A \cap B$
- Reverse: $A^R = \{ a_1 a_2 ... a_n \mid a_n ... a_1 \in A\}$

Theorem: The class of regular languages is closed under all six of these operations

Proving perties

Complement

Complement: $\bar{A} = \{ w \mid w \notin A\}$

Theorem: If A is regular, then $\bar{A}$ is also regular

Proof idea:

Complement, Formally

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA recognizing a language A. Which of the following represents a DFA recognizing $\bar{A}$?

a) $(F, \Sigma, \delta, q_0, Q)$

b) $(Q, \Sigma, \delta, q_0, Q$ he set of states in Q that are not in F

c) $(Q, \Sigma, \delta', q_0, F)$ where $\delta'(q,$ uch that $\delta(p, s) = q$

d) None of the above

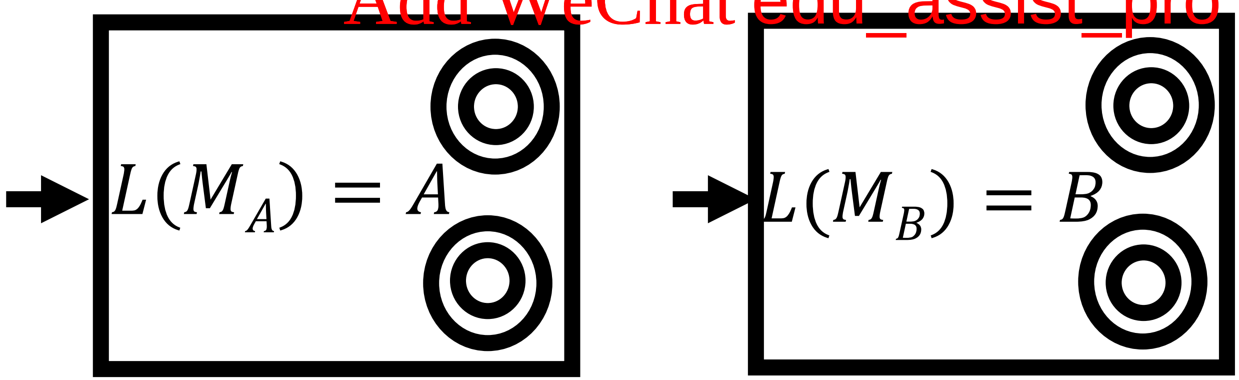
Closure under Concatenation

Concatenation: $A \circ B = \{ xy \mid x \in A, y \in B \}$

Theorem. If A and B are regular, $A \circ B$ is also regular.

Proof idea: Given DFAs M_A and M_B, construct NFA by

- Connecting all acc rt state in M_B.
- Make all states in



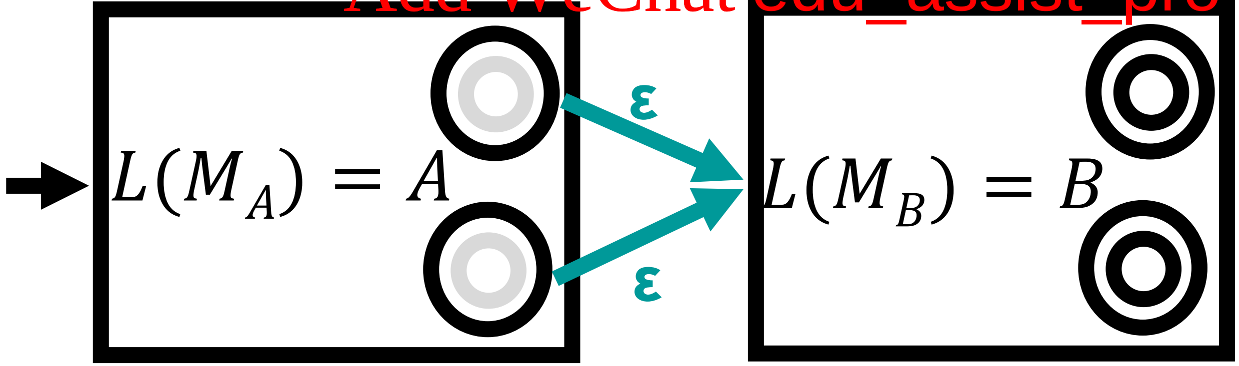
Closure under Concatenation

Concatenation: $A \circ B = \{ xy \mid x \in A, y \in B \}$

Theorem. If A and B are regular, $A \circ B$ is also regular.

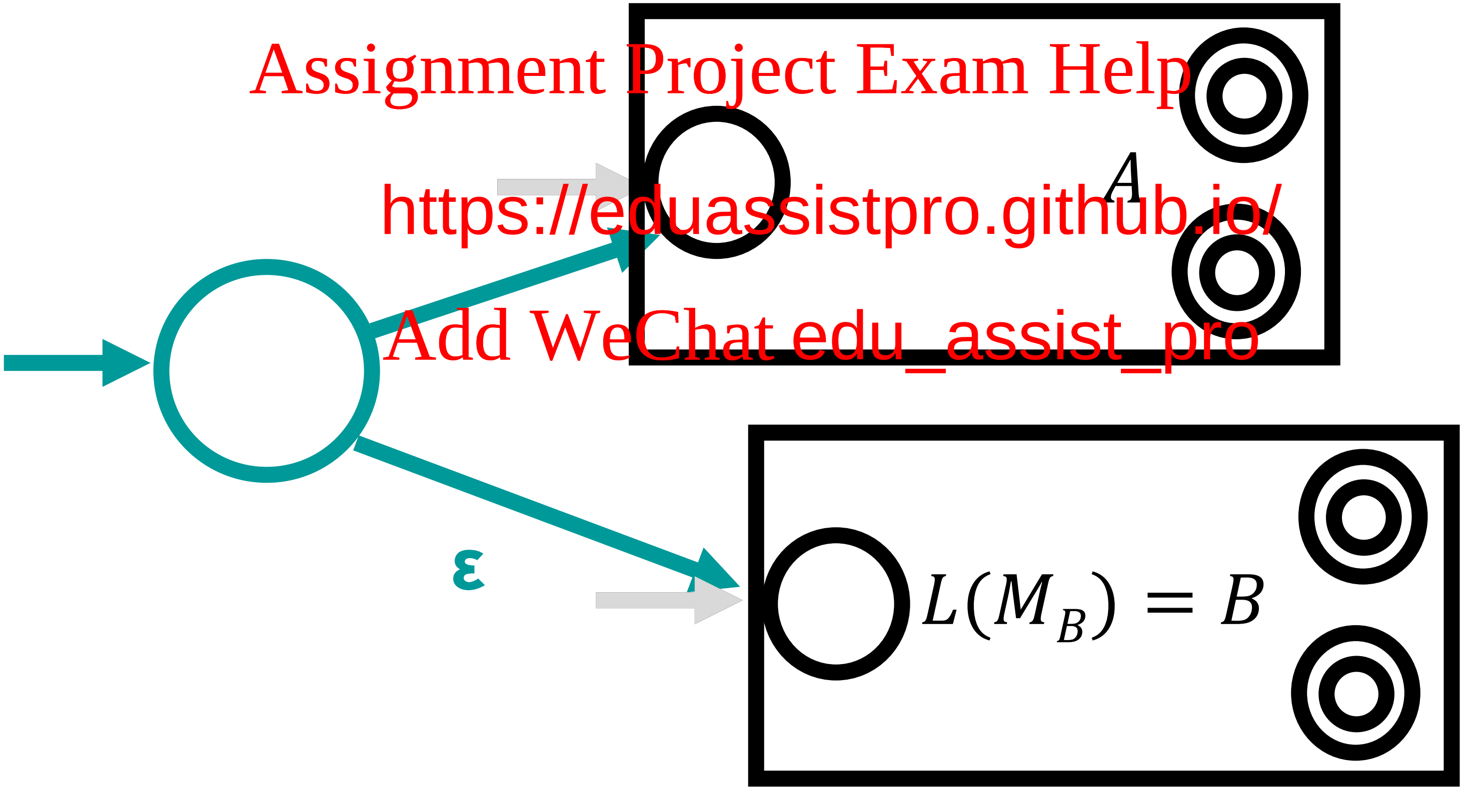
Proof idea: Given DFAs M_A and M_B, construct NFA by

- Connecting all acc rt state in M_B.
- Make all states in



A Mystery Construction

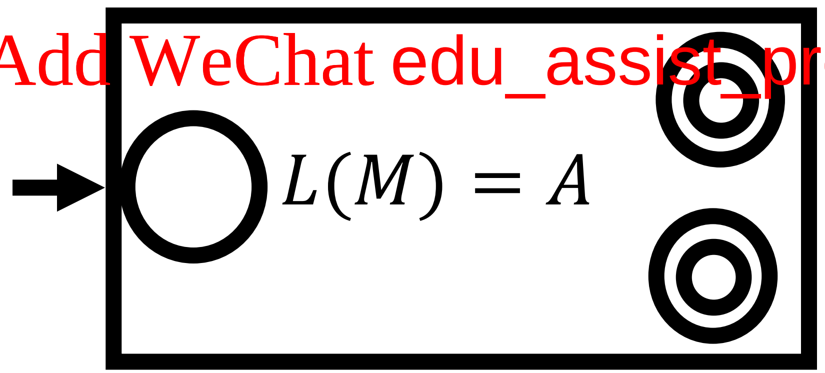
Given DFAs M_A recognizing A and M_B recognizing B, what does the following NFA recognize?



Closure under Star

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

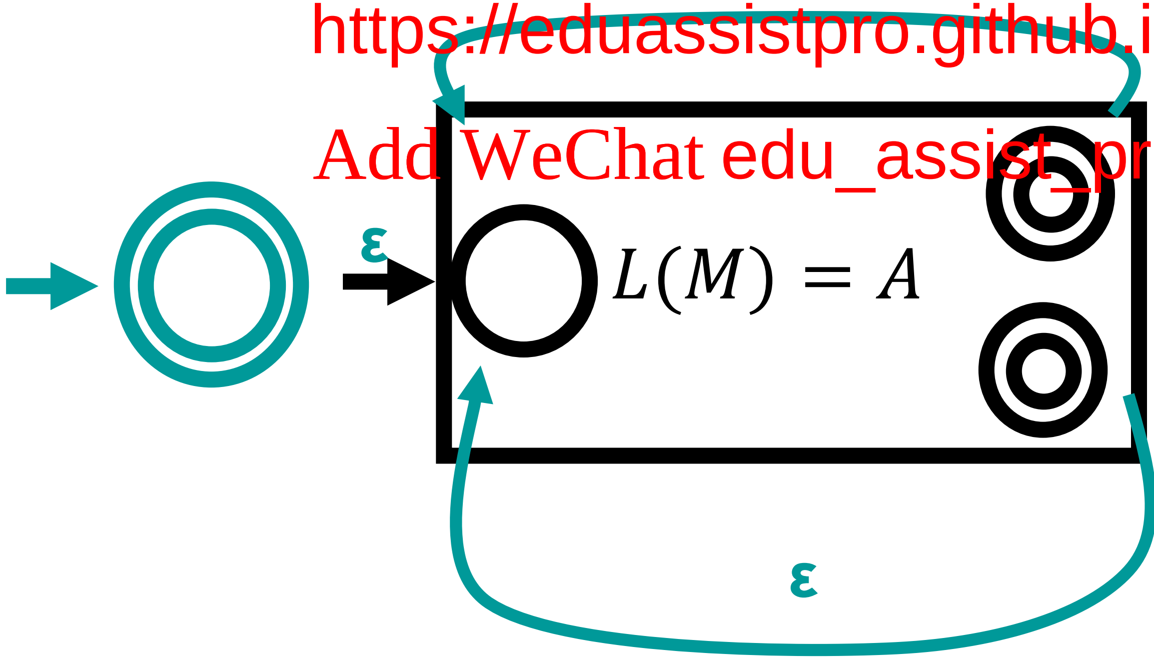
Theorem. If A is regular, A^* is also regular.



Closure under Star

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

Theorem. If A is regular, A^* is also regular.



On proving your own closure properties

You'll have homework/test problems of the form "show that the regular languages are closed under some operation"

What would Sipse

- Give the "proof i ake machine(s) recognizing regular language(s te a new machine

- Explain in a few sentences why the construction works
- Give a formal description of the construction
- No need to formally prove that the construction works

Regula ns

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: $\emptyset, \{\varepsilon\}, \{a\}$ $\in \Sigma$

Regular operations:

Union: $A \cup B$

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

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

Regular Expressions – Syntax

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

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

2. If R_1 and R_2 are regular exp hen 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

1. $L(\emptyset) = \emptyset$
2. $L(\varepsilon) = \{\varepsilon\}$
3. $L(a) = \{a\}$
4. $L((R_1 \cup R_2)) = L(R_1) \cup L($
5. $L((R_1 \circ R_2)) = L(R_1) \circ L(\;_2$
6. $L\big((R_1^*)\big) = (L(R_1))^*$

Example: $L(((a^*) \circ (b^*))) =$

Simplifying Notation

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

- 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

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

Examples

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

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

2. $\{w \mid w$ has length at least 3 rd symbol is $0\}$

3. $\{w \mid$every odd position of w is $1\}$

Syntactic Sugar

- For alphabet Σ, the regex Σ represents $L(\Sigma) = \Sigma$

- For regex R, the $+$ $*$

Regexes in the Real World

`grep` = globally search for a regular expression and print matching lines

Not captured by regular expressions: Backreferences

Equivalence of Regular Expressions 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 e egular 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

What should the inductive hypothesis be?

a) Suppose **som** length k can be converted to

b) Suppose **every** regular expr ength k can be converted to an NFA

c) Suppose **every** regular expression of length **at most** k can be converted to an NFA

d) None of the above

Regular expression -> NFA

Theorem 1: Every regex has an equivalent NFA

Proof: Induction on size of a regex

Inductive step:

$R = (R_1 \cup$

$R = (R_1 R_2)$

$R = (R_1^*)$

Example

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