CSE 105
THEORY OF COMPUTATION

"Winter" 2018

http://cseweb.ucsd.edu/classes/wi18/cse105-ab/
Today's learning goals

• Convert between regular expressions and automata
• Describe the limits of the class of regular languages

Announcements:

- Group HW2 due Saturday
- Review Quiz 3 available
- Indiv HW3, Group HW3 available
- Office Hours: see Calendar
DFA equiv NFA equiv RegExp

**Theorem:** For each language L,

L is recognizable by some DFA iff L is recognizable by some NFA iff L is describable by some regular expression.
From RegExp to DFA

Structural induction!

- Build DFAs (or NFAs) corresponding to base cases in inductive definitions of regular expressions.

- Describe constructions for DFAs corresponding to each of the inductive steps: union, concatenation, Kleene star.
Structural induction

Base cases

Which of these recognizes $L(\emptyset)$ ?

A. NFA #1
B. NFA #2
C. NFA #3
D. More than one of the above
E. None of the above

$\emptyset = \{\varepsilon\}$

Thm 1.45, 1.47, 1.49 pp 59-62
Structural induction

• Base cases

\[ L(\varepsilon) = \]

\[ L(\epsilon) = \]

\[ L(\emptyset) = \]

\[ L(\varepsilon) = \]

\[ L(\emptyset) = \]

\[ L(\varepsilon) = \]

• Inductive steps

Input \[ \varepsilon \] M1

Union

Input \[ \varepsilon \] M2

Concatenation

Kleene star
Example

\[ a^* (ab)^* = a^* \cdot (ab)^* \]

\[
\text{NFA for } L(a)
\]

\[
\text{NFA for } L(a^*)
\]

\[
\text{NFA for } L((ab)^*)
\]

\[
\text{NFA for } L(a^*(ab)^*)
\]
Example

\( a^* U b^* \)
From DFA to RegExp

Trace possible paths from start state to accept state.

Intermediate machines can have regular expressions on transitions.

First:
1. add new start state that has $\varepsilon$ arrow to old start state
2. add new accept state that has $\varepsilon$ arrows from old accept states (and $\emptyset$ arrows from nonaccept states)
no longer accept states
From DFA to RegExp

Remove one state at a time.

- Restore automaton by modifying regular expressions on transitions that went through removed state.
Get: $\varepsilon \cup a^*u^*b^*$

Compare: $\varepsilon \cup b^*a^*u^*$
Regular languages

To prove that a set of strings is regular:

1. Build a **DFA** whose language is this set.  OR
2. Build an **NFA** whose language is this set.  OR
3. Build a **regular expression** describing this set.  OR
4. Use the **closure** properties of the class of regular languages to construct this set from others known to be regular (**complementation**, **union**, **intersection**, **concatenation**, **Kleene star**, **flipbits**, **reverse**, etc.)
New closure operations

• Strategy: want to show class of regular languages is closed under some operation

• Start with DFA M1, …, M2

• Design NFA that recognizes OP( L(M1), …, L(M2) )
All roads lead to … regular sets?

Are there any languages over \{0,1\} that are not regular?

A. Yes: a language that is recognized by an NFA but not any DFA.

B. Yes: there is some finite language of strings over \{0,1\} that is not described by any regular expression.

C. No: all languages over \{0,1\} are regular because that's what it means to be a language.

D. No: for each set of strings over \{0,1\}, some DFA recognizes that set.

E. None of the above.
Counting languages

How many languages over \{0,1\} are there?

A. Finitely many because \{0,1\} is finite.
B. Finitely many because strings are finite.
C. Countably infinitely many because \{0,1\}^* is countably infinite.
D. Uncountably many because languages are in the power set of \{0,1\}^*.
E. None of the above.
Counting regular languages over \{0,1\}:
\[|\{\text{regular languages}\}| \leq |\{\text{regular expressions}\}|\]

Each regular expression is a finite string over the alphabet:
\[
\{0, 1, \varepsilon, \emptyset, (, ), U, *\}
\]

The set of strings over an alphabet is countably infinite.

Conclude: countably infinitely many regular languages.
Where we stand

Fact 1: There exist nonregular languages.

Fact 2: If we know some languages are nonregular, we can conclude others must be too. (cf. Discussion)

*But, we don't have any specific examples of nonregular languages.*

*Yet.*
For next time

- Work on Group Homework 2 due Saturday

Pre class-reading for Friday: page 77.