CSE 105
THEORY OF COMPUTATION

Fall 2021

http://cseweb.ucsd.edu/classes/fa21/cse105-a/
Today's learning goals

• Are there non-regular languages?
• Pumping Lemma for Regular Languages
• Identify some nonregular sets
"Regular = regular"  Sipser Theorem 1.54 p. 66

**Theorem:** A language is regular if and only if some regular expression describes it.

**Lemma 1.55:** If a language is described by a regular expression, then it is regular.

**Lemma 1.60:** If a language is regular, then it is described by some regular expression.
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 infinite 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. I don't know.
Proving nonregularity

How can we prove that a set is non-regular?

A. Try to design a DFA that recognizes it and, if the first few attempts don't work, conclude there is none that does.
B. Prove that it's a strict subset of some regular set.
C. Prove that it's the union of two regular sets.
D. Prove that its complement is not regular.
E. I don't know.
Counting

• **Fact**: the set of real numbers is uncountable
• **Fact**: the set of subsets of \( \{0,1\}^* \) is uncountable.

• **Fact**: there are countably many DFA with \( \Sigma=\{0,1\} \)
• **Fact**: there are countably many regular languages over \( \{0,1\} \)
Counting

- Fact: the set of real numbers is uncountable.
- Fact: the set of subsets of \{0,1\} is uncountable.
- Fact: there are countably many DFA with \(\Sigma = \{0,1\}\).
- Fact: there are countably many regular languages over \{0,1\}.

Uncountably many languages over \{0,1\}

Countably many regular languages over \{0,1\}
Where we stand

• There exist non-regular sets.

• If we know that some sets are not regular, we can conclude others are also not regular judiciously reasoning using closure properties of class of regular languages.

• No example of a specific regular set ... yet.
Diagonalization

• Let $R$ be the set of syntactically valid regular expressions over binary alphabet $\{0,1\}$
  - $R$ is a language over the alphabet $\Sigma = \{0,1,\varepsilon,{},\cup,(),^*\}$
  - Encode $\Sigma$ as 3-bit “bytes”: 000,001,010,…,111

• For each regular expression $E$ in $R$ we have
  - A string $\text{encode}(E)$ in $\{0,1\}^*$
  - A set of strings $L(E) \subset \{0,1\}^*$

• Let $D = \{ \text{encode}(E) : \text{encode}(E) \text{ is not in } L(E) \} \subset \{0,1\}^*$

• Challenge Question: is $D$ regular?
DFAs and Counting

Which of the following languages is regular?

- $L_1 = \{0^n1^n \mid n < 10\}$
- $L_2 = \{0^n1^n \mid n > 10\}$

A) $L_1$
B) $L_2$
C) Both $L_1$ and $L_2$
D) Neither $L_1$ nor $L_2$
E) I don’t know
Which of the following languages is regular?

- $L_1 = \{0^n1^n \mid n < 10\}$
- $L_2 = \{0^n1^n \mid n > 10\}$

Correct answer is (A): Only $L_1$

Why is $L_1$ regular?
- Easy: because it is a finite language (of size 10)

But why is $L_2$ not regular? Definitions look so similar!
A DFA for L1

• Give a DFA for
  - \( L_1 = \{0^n1^n \mid n < 10\} \)

• How many states do you need?

A) 10 (or fewer)
B) 20 (or fewer)
C) 100 (or fewer)
D) Exactly 2048
E) I don’t know
Let’s solve it in JFLAP (n<5)
One more question

• Which of the following languages are regular?

  - $L_1 = \{0^n1^n \mid n \geq 0 \}$
  - $L_2 = \{0^n1^n \mid (5<n) \text{ and } (n<10)\}$
  - $L_3 = \text{Complement of } L_2$
  - $L_4 = \{0^n1^n \mid \text{not}(5<n) \text{ or not}(n<10)\}$

A) All four  
B) L2, L3  
C) L2, L3, L4  
D) None  
E) I don’t know
Intuition

• DFAs can count up to 10
• DFAs can count up to any fixed bounded number k, given enough states.
• But they cannot count indefinitely:
  - Given a long enough string 0000000000...0, they will lose count when they “run out” of states
• How can we formalize this into a proof?
Bounds on DFA

• in DFA, memory = states

• Automata can only "remember"...
  • ...finitely far in the past
  • ...finitely much information

• If a computation path visits the same state more than once, the machine can't tell the difference between the first time and future times it visited that state.
A non-regular language

• Proof that $L = \{0^n1^n \mid n \geq 0\}$ is not regular
• Proof: assume $L$ is regular (for contradiction)
  – $L = L(M)$ is the language of a DFA $= (Q, \ldots)$ with a finite number of states $p = |Q|$
  – Consider computation on input $0^p1^p$
  – Computation has length $2p+1 > |Q|$
  – Must visit the same state twice
  – This gives many other accepting computations
Proof details

- Assume \( L(M) = \{0^n1^n \mid n \geq 0\} \)
- \( M \) has \( p=|Q| \) states and accepts \( 0^p1^p \)
- \( M \) must visit the same state twice while reading \( 0^p \)
- Break the computation/input in three parts
  - Accepts not only 000000111111
  - But also 00111111, 000000000111111
  - \( 00(000)^k01111111 \)
- Contradicts the definition of \( L(M) \)
Pumping Lemma: What and Why

- Pumping lemma abstracts this pattern of reasoning to prove that a language is not regular.
- Pumping Lemma: asserts a property satisfied by all regular languages.
- Using the pumping lemma
  - Assume (for contradiction) that L is regular.
  - Therefore it satisfies pumping property.
  - Derive a contradiction.
Pumping Lemma: informal

• All sufficiently long strings accepted by a DFA can be pumped
Pumping

• Focus on computation path through DFA

Idea: if one long string $xyz$ is accepted, then many other strings have to be accepted too: $xz, xyz, xyyz, xyyyz, ...$
Pumping Lemma

If $A$ is a regular language, then there is a number $p$ (*the pumping length*) where, if $s$ is any string in $A$ of length at least $p$, then $s$ may be divided into three pieces, $s = x y z$ such that

- $|y| > 0$, and
- for each $i \geq 0$, $x y^i z \in A$,
- $|xy| \leq p$. 

*Sipser p. 78 Theorem 1.70*