YOUR PROJECTS ARE DUE TODAY BY 5:00 PM. I DIDN’T EVEN KNOW WE HAD ONE.

WAIT. I DON’T THINK I’VE BEEN ATTENDING. I MUST HAVE FORGOTTEN I HAD THIS CLASS. SHITSHITSHIT.

OKAY, I’M GONNA FAIL. WILL IT HOLD ME BACK? I JUST WANT TO GET OUT OF HERE. I THOUGHT I’D FINISHED MY REQUIREMENTS ALREADY.

IN FACT, I THINK I REMEMBER GRADUATING.

WHAT THE HELL IS-

FUN FACT: DECADES FROM NOW, WITH SCHOOL A DISTANT MEMORY, YOU’LL STILL BE HAVING THIS DREAM.
irregular languages

Let \( \Sigma = \{1, +, -\} \).
Define \( L_1 = \{x + y = z : x, y, z \in \{1\}^* \text{ and } (x)_1 + (y)_1 = (z)_1\} \). That is, \( L_1 \) is all the strings of true statements of the form \( x + y = z \) where + and = are characters and \( x, y, \) and \( z \) are strings of 1’s interpreted as unary numbers. For example, “111 + 11 = 11111” \( \in L_1 \), because \( 3 + 2 = 5 \).
But, “111 + 11 = 11” \( \notin L_1 \), because \( 3 + 2 \neq 2 \).

Prove that \( L_1 \) is not a regular language.

\[ L \text{ is regular} \implies \text{accepted by some DFA } M \]
Prefixes = \{ 1, 11, 111, 1111, \ldots \} 3
\[
\exists s_i, s_j \text{ that lead to the same state of } M
\]
\[ s_i \rightarrow 1's, \quad s_j \rightarrow 0, \quad i < j \]
\[ s_i + 1 = 11\ldots1 \in L_1 \quad \& \quad s_j + 1 = 11\ldots1 \text { invalid } \quad \Rightarrow \quad M \text{ does not recognize } L \]
\[ \Rightarrow \quad L_1 \text{ is irregular} \]
irregular languages

Let $\Sigma = \{0, 1, +, =\}$.
Define $L_2 = \{x + y = z : x, y, z \in \{0, 1\}^* \text{ and } (x)_2 + (y)_2 = (z)_2\}$. That is, $L_2$ is the same idea as $L_1$, except the numbers are interpreted as binary instead of unary. For example, “111 + 11 = 1000” $\in L_2$, because $5 + 3 = 8$. But, “111 + 11 = 11” $\not\in L_2$, because $7 + 3 \neq 3$.

Prove that $L_2$ is not a regular language.

Prefixes $= \{0, 1^3, 1^4, \ldots\}$ $\not\subseteq \Sigma^*$

Prefixes $\not\subseteq \{0, 1^3, 1^4, \ldots\}^*$

$S_i, S_j$ end at same state

$s_i + 1 = (s_i + 1)_2 \in L_2$

$s_i + 1 = (s_i + 1)_2 \not\in L_2$

$\Rightarrow L_2$ is irregular.
convert NFA to DFA

Diagram of a non-deterministic finite automaton (NFA) with states and transitions. The NFA is converted to a deterministic finite automaton (DFA) as shown in the diagram.
design an NFA

The set of all binary strings that start with two one's and end with two one's.
design an NFA

The set of all binary strings that start with two one's or end with two one's.

(See prev. slide)
The set of all binary strings that are of odd length and have 1 as their middle character.

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

$$S \rightarrow 1 \mid 0S0 \mid 1S1 \mid 0S1 \mid \text{same length.}$$

Example:

$$S \rightarrow 0S1 \rightarrow 01S11 \rightarrow 011111.$$
All binary strings that contain at least two 0’s and at most two 1’s.
Let $A$ be a set. Let $R$ and $S$ be transitive relations on $A$.

(a) Is $R \cup S$ necessarily transitive? Prove your answer.

(b) Is $R \cap S$ necessarily transitive? Prove your answer.

(a)

\[ (a,c) \in R \quad \text{trans.} \]
\[ (b,c) \in S \quad \text{trans.} \]
\[ R \cup S \text{ not trans.} \]

(b) Yes.

\[ (b,c), (a,b) \in R \cap S \]
\[ \Rightarrow (a,c) \in R \quad \text{by trans.} \]
\[ (a,c) \in S \quad \text{by trans.} \]
\[ \Rightarrow (a,c) \in R \cap S. \]
We use \( \mathbb{Z}^+ \) to mean the set of positive integers. Let \( R \subseteq (\mathbb{Z}^+ \times \mathbb{Z}^+) \times (\mathbb{Z}^+ \times \mathbb{Z}^+) \) be the relation given by \(((a, b), (c, d)) \in R\) if and only if \(ad = bc\). Prove that \( R \) is reflexive, symmetric and transitive.

**Reflexive:** \( (a, b), (a, b) \in R \) \( \forall (a, b) \in A \)

\[
ab = ba
\]

**Symmetric:** \( (a, b), (c, d) \in R \Rightarrow (c, d), (a, b) \in R \)

\[
\iff ad = bc \Rightarrow cb = da
\]

**Transitive:** \( (a, b), (c, d) \in R \) \( \Rightarrow (c, d), (e, f) \in R \)

\[
\Rightarrow (a, b), (e, f) \in R \iff \frac{ad = b}{c} = \frac{ef}{d} = e
\]

\[
\iff ad = bc \land cf = de \Rightarrow af = be
\]
Consider the following DFA, $M$:

For each of the four states, $q$, in $M$, write a regular expression that matches exactly the strings that end at the state $q$ when starting from the initial state.
Let $c > 0$ be an integer. The following recursive definition describes the running time of a recursive algorithm.

\[
T(0) = 0 \\
T(n) \leq c \\
T(n) = T\left(\left\lfloor \frac{3n}{4} \right\rfloor \right) + T\left(\left\lfloor \frac{n}{5} \right\rfloor \right) + cn \\
\text{for all } n \leq 20 \\
T(n) = \text{ct}_{\left(\left\lfloor \frac{3(n+1)}{4} \right\rfloor \right)} + \text{ct}_{\left(\left\lfloor \frac{n+1}{5} \right\rfloor \right)} + c(n+1) \\
\text{for all } n > 20
\]

Prove by strong induction that $T(n) \leq 20cn$ for all $n \geq 0$.

\[
P(n) = " T(n) \leq 20cn "
\]

Base case: $T(0) = 0 \leq 0 = 20c \cdot 0$.

For $n \leq 20$, $T(n) \leq c \leq 20c \cdot n$.

Ind. hypothesis: For some $n > 20$, $P(0), \ldots, P(n)$ hold.

Ind. step: $T((n+1)) = T\left(\left\lfloor \frac{3(n+1)}{4} \right\rfloor + T\left(\left\lfloor \frac{n+1}{5} \right\rfloor \right) + c(n+1)$

\[
\leq 20c \left(\frac{3(n+1)}{4}\right) + 20c \left(\frac{n+1}{5}\right) + c(n+1)
\]
Define $f_n$ and $g_n$ as follows for $n \in \mathbb{N}$:

\[
\begin{align*}
    f_0 &= 1 \\
    f_1 &= 5 \\
    f_2 &= 10 \\
    f_n &= 2f_{n-1} - 4f_{n-2} \text{ for } n \geq 3
\end{align*}
\]

\[
\begin{align*}
    g_0 &= 1 \\
    g_1 &= 5 \\
    g_2 &= 10 \\
    g_3 &= 0 \\
    g_4 &= -40 \\
    g_n &= 2g_{n-1} - 3g_{n-2} - 2g_{n-3} + 4g_{n-4} \text{ for } n \geq 5
\end{align*}
\]

Prove that $f_n = g_n$ for all $n \in \mathbb{N}$. 
Give a recursive definition of the functions max and min so that \( \max(a_1, a_2, \ldots, a_n) \) and \( \min(a_1, a_2, \ldots, a_n) \) are the maximum and minimum of the \( n \) numbers \( a_1, a_2, \ldots, a_n \), respectively.

\[
\max(a_1) = a_1
\]

\[
\max(a_1, \ldots, a_n) = \max(a_1, \max(a_2, \ldots, a_n))
\]

\[
\max(a_1, a_2) = \max(a_1, \max(a_2)) = \max(a_1, a_2)
\]

\[
\max(a_1, a_2) = \begin{cases} a_1 & a_1 > a_2 \\ a_2 & a_1 \leq a_2 \end{cases}
\]
recursive definitions

Use structural induction to show that $l(T)$, the number of leaves of a full binary tree $T$, is 1 more than $i(T)$, the number of internal vertices of $T$.

**Base Case ($B$):**

- $T = \cdot$
  - $B: l(\cdot) = 1$

**Recursive Case ($R$):**

- $T_1, T_2$
  - $R: l(T) = l(T_1) + l(T_2)$

**Base Case ($B$):**

- $i(\cdot) = 0$

**Recursive Case ($R$):**

- $T_1, T_2$
  - $R: i(T) = 1 + i(T_1) + i(T_2)$
Use structural induction to show that $l(T)$, the number of leaves of a full binary tree $T$, is 1 more than $i(T)$, the number of internal vertices of $T$.

\[ l(T) = 1 + i(T) \quad \forall T \]

\[ P(T) = \left\{ l(T) = 1 + i(T) \right\} \]

**Pf by struct. ind.**

**Base:** $l(\emptyset) = 1 = 1 + 0 = 1 + i(\emptyset)$. \[ l(\emptyset) \quad \text{and} \quad l(T) \quad \text{hold} \]

**Ind Hypothesis:** $P(T_1)$ and $P(T_2)$ hold

\[ l(T) \quad \text{def} \quad l(T_1) + l(T_2) = 2 + i(T_1) + i(T_2) \]

\[ = 1 + (1 + i(T_1) + i(T_2)) \]

**By str. ind. \( \forall T \ P(T) \).**
- If a set $A$ is countable, then every subset of $A$ is countable.  
  \text{False. Subset could be finite.}

- If a $L$ is generated by a context-free grammar, then every subset of $L$ is generated by a context-free grammar.  
  \text{False. $L = \Sigma^*$ so any lang. is a subset $L$.}

- There is a Java program that takes $(P, x)$ as input and decides whether $P$ halts on $x$ within $2^{|x|}$ steps where $|x|$ is the length of $x$.  
  \text{True (just do it)}

- If $P(x)$ is true for some $x$ in the domain and false for others, and $Q(x)$ is always true, then $\exists x \ (P(x) \rightarrow Q(x))$ is true.  
  \text{True.}
- If $R$ and $R'$ are relations on the same set and $R \subseteq R'$ then $R'$ reflexive $\Rightarrow$ $R$ reflexive. 

False.

- A relation $R$ is anti-reflexive if $(x, x) \notin R$ for every $x$. If $R$ is anti-reflexive, then $R^2$ is anti-reflexive.

$$R = \{(a, x), (x, a)\}$$

$$R^2 = \{(a, a), (x, x)\}$$

So False.

- If $L$ is regular, then the language $\{xx : x \in L\}$ has a CFG.

$$S \rightarrow XX$$

$$X \rightarrow \ldots$$

$\{x_1x_2 : x_1, x_2 \in L\}$

True.

- Every regular language is decidable.