Circuit-Equiv: \( \{(a, b) \mid a \text{ encodes a circuit equiv to the circuit } b \text{ encodes } \} \)

countable infinite representations of languages

but

uncountable \# of languages - suppose we list languages as \( \{0, 1\} \)

\[ L_1: \begin{array}{ccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} \]
\[ L_2: \begin{array}{cccc} 1 & 1 & 1 & 1 \end{array} \]
\[ L_3: \begin{array}{ccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} \]

which languages can we represent?
Using $\cup$, $\cdot$, $^*$, $\omega$, ... what languages can we represent?

Can we represent \( \{ w \in \{0, 1\}^* \mid w \text{ has exactly two } 1\text{'s} \} \)

\( \{03^* \cdot 1 \cdot 03^* \cdot 1 \cdot 03^* \} \)

\( \{ w \mid w \text{ has at most one } 1 \} \)

\( \{03^* \cup 003^* \mid 03^* \} \)

\( \{ w \mid w \text{ begins and ends with } 1 \} \)

\( \{1 \}^* \cdot 0 \cdot 1^* \cdot 0 \cdot 1^* \cdot \{1 \} \)

\( \{ w \mid \text{ same # of } 0\text{'s as } 1\text{'s} \} \)

Can't do this one!
Regular expressions: simpler representation of

Formally, for alphabet \( \Sigma \),

- \( \emptyset \) is a regular expression
- \( a \) is a regular expression for all \( a \in \Sigma \)
- if \( \alpha, \beta \) are regular expressions, so are

\[
\alpha \beta, \\
\alpha \cup \beta, \\
\alpha^*
\]

Regular expressions are representations of the language they generate

- \( \emptyset \) represents \( \emptyset \)
- \( \{a\} \) represents \( \{a\} \)
- \( \{ab\} \) represents \( \{ab\} \)
- \( \{a\} \cup \{b\} \) represents \( \{a\} \cup \{b\} \)
- \( \{ab\} \) represents \( \{ab\} \)
- \( \{a\}^* \) represents \( \{a\}^* \)
- \( \{a\} \) represents \( \{a\} \)
- \( \{ab\} \) represents \( \{ab\} \)
\( a(a^*b^*)a \) represents ??

\[ \{a\} \cup \{a^*b^*\} \cup \{a\} = \{a\} \cup \{a^*b^*\} \cup \{a\} \]

\[ \uparrow \]

\[ L(ab) \cup L(b) \]

\[ \{ \text{words, or more } a\text{'s} \} \quad \text{followed by any # of } b\text{'s} \]

\[ \{a\}^* \cup \{b\}^* \]

\[ \{a^*b^* \cup \{a\} \}^* \]

\[ L((a \cup ab)^*) \]

\[ = L((a \cup ab)^*) \]

\[ = L((a \cup ab)^*) \]

\[ = (L(a) \cup L(ab))^* \]

\[ = (\{a\} \cup \{a,b\})^* \]

\[ = \{a,ab\}^* \quad \text{if every } b \text{ is preceded by an } a \]

Regular expression for \( \{w \in \{a, b\}^* \mid w \text{ has even # of } a\text{'s} \} \)
\\[ \{ w \mid w \text{ has exactly 2 a's} \} \]

\[(b^*ab^*a^*)^* a^*b^*b^*a^* \]

Regular language: language represented by some regular expression

What languages aren't regular?

Deterministic Finite Automaton (DFA)

- A machine that, given a string, outputs \( Y \) or \( N \)
- Input tape
- Start state
- Read head
- Finite state control

Starts at beginning of input

On each step, change state based on:
- Current symbol
- Current state

At end of input, \( Y \) or \( N \) depending on state
Formally, a DFA is $\langle K, \Sigma, \delta, s, F \rangle$ where:

- $K$ is a subset of $K$ of accepting states
- $\Sigma$ is the alphabet
- $s$ is the start state
- $F$ is the set of states
- $\delta: (K \times \Sigma) \rightarrow K$ is the transition function

Given:

$K = \{ s_0, s_1 \}$
$\Sigma = \{ a, b \}$

Transition table:

<table>
<thead>
<tr>
<th></th>
<th>$a$</th>
<th>$b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_0$</td>
<td>$s_0$</td>
<td>$s_1$</td>
</tr>
<tr>
<td>$s_1$</td>
<td>$s_1$</td>
<td>$s_0$</td>
</tr>
</tbody>
</table>

New $K$: $s = s_0$, $F = \{ s_0 \}$

The DFA accepts $aabaabaa$, rejects $aba$, and accepts $\{ w | w \text{ has even length} \}$.