Closure properties of regular languages

Proposition:
Let \( \Sigma \) be some finite alphabet.
If \( L \subseteq \Sigma^* \) is regular, then so is \( \overline{L} = \Sigma^* \setminus L \).

Proof:
Closed under union

**Theorem:**
Let $\Sigma$ be some finite alphabet.
If $L_1 \subseteq \Sigma^*$ and $L_2 \subseteq \Sigma^*$ are regular, then so is $L_1 \cup L_2$.

**Proof:**

The mindset

**Step 1:** Imagining ourselves as a DFA
Closed under union

**Example**

$L_1 = \text{strings with even number of 1's}$

$L_2 = \text{strings with length divisible by 3}$. 

**Input:** 101001

Accept
Closed under union

Main idea:

Step 2: Formally defining the DFA
### Closed under union

**Proof:** Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA deciding $L_1$ and $M' = (Q', \Sigma, \delta', q'_0, F')$ be a DFA deciding $L_2$. We construct a DFA $M'' = (Q'', \Sigma, \delta'', q''_0, F'')$ that decides $L_1 \cup L_2$, as follows:

### More closure properties

- **Closed under union:**

- **Closed under concatenation:**

- **Closed under star:**

### super awesome vs regular

What is the relationship between super awesome and regular?
super awesome vs regular

Theorem:
Can define regular languages recursively as follows:

Closed under concatenation

Theorem:
Let $\Sigma$ be some finite alphabet.
If $L_1 \subseteq \Sigma^*$ and $L_2 \subseteq \Sigma^*$ are regular, then so is $L_1L_2$.

The mindset

Imagine yourself as a DFA.

Rules:
1) Can only scan the input once, from left to right.
2) Can only remember “constant” amount of information.

should not change based on input length
**Step 1**: Imagining ourselves as a DFA

Given $w \in \Sigma^*$, we need to decide if
$$w = uv \text{ for } u \in L_1, v \in L_2.$$  

**Problem**: Don’t know where $u$ ends, $v$ begins.  
When do you stop simulating $M_1$ and start simulating $M_2$?

---

Suppose you know $u$ ends at $w_3$.

|     | ... | 1 | 0 | 0 | 1 | 0 | 0 | 1 | ...
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_1$</td>
<td>$w_2$</td>
<td>$w_3$</td>
<td>$w_4$</td>
<td>$w_5$</td>
<td>$w_6$</td>
<td>$w_7$</td>
<td>$w_8$</td>
<td>$w_9$</td>
<td>$w_{10}$</td>
</tr>
</tbody>
</table>

| $q_0$ | $q_1$ | $q_1$ | $q_2$ | $q_2$ | $q_2$ | $q_1$ | $q_2$ | $q_2$ | $q_1$ |

**thread:**
Step 2: Formally defining the DFA

\[ M_1 = (Q, \Sigma, \delta, q_0, F) \quad M_2 = (Q', \Sigma, \delta', q'_0, F') \]

\[ Q'' = \]

\[ \delta'' : \]

\[ q''_0 = \]

\[ F'' = \]