**THM.** Every string is perfectly cromulent.

**Proof:** Let \( w \) be an arbitrary string. Assume, for every string \( x \) such that \( |x| < |w| \), that \( x \) is perfectly cromulent. There are two cases to consider.

- Suppose \( w = \varepsilon \).

Therefore, \( w \) is perfectly cromulent.

- Suppose \( w = ax \) for some symbol \( a \) and string \( x \). The induction hypothesis implies that \( x \) is perfectly cromulent.

Therefore, \( w \) is perfectly cromulent.

In both cases, we conclude that \( w \) is perfectly cromulent.

**Lemma:** For all strings \( w, y, z \),

\[
(w \cdot y) \cdot z = w \cdot (y \cdot z)
\]

**Proof:** Let \( w, y, z \) be arbitrary strings. Assume \( (x \cdot y) \cdot z = x \cdot (y \cdot z) \) for all \( x \) s.t. \( |x| < |w| \), all \( y \), all \( z \).

**Case 1:** \( w = \varepsilon \)

\[
(w \cdot y) \cdot z = (\varepsilon \cdot y) \cdot z = \varepsilon \cdot (y \cdot z) = w \cdot (y \cdot z) \quad \text{[\( w = \varepsilon \)]}
\]

**Case 2:** \( w = ax \) for some symbol \( a \), string \( x \).

\[
(w \cdot y) \cdot z = (a \cdot x) \cdot y \cdot z = a \cdot (x \cdot y) \cdot z = a \cdot ((x \cdot y) \cdot z) \quad \text{[\( w = ax \)]}
\]
In both cases, \((w \cdot y) \cdot z = w \cdot (y \cdot z)\)

**LANGUAGES**

Language: set of strings over an alphabet

\(\Sigma = \{0, 1, 2\}\)

Examples of languages:

- \(\emptyset\) → EMPTY SET (no strings)
- \(\{\varepsilon\}\) → set containing the empty string

\(\emptyset\) \(\varepsilon\) NOT A LANGUAGE

\(\Sigma^*\): All strings over \(\Sigma\)

\(S^*\): All strings formed by concatenating symbols from set \(S\).

\(\Sigma^5\): All strings of length 5 formed by concatenating symbols from \(\Sigma\).
\[ \Sigma = \{ \Sigma A, B, C, \ldots \} \]

\[ \exists x \in \Sigma \text{ is a language} \]

\[ L = L_1 \cup L_2 \]

\[ L = L_1 \cap L_2 \]

\[ L = \overline{A} = \Sigma^* \setminus A \]

\[ L = \text{All python programs} \]

\[ L = A \circ B = \{ x \circ y | x \in A, y \in B \} \]

\[ \{ \text{OVER, UNDER} \} \circ \{ \text{EAT, PAY} \} \]

\[ \Sigma \{ \text{a}, \text{b}, \text{c} \} \]

\[ \phi \circ L = \phi \quad \exists \in \Sigma^* \circ L = L \]

Also,

\[ L^* = \exists \in \Sigma^* \cup \cup L \cup \cup \ldots \]

\[ w \in L^* \iff w = \varepsilon \text{ or } w = x \circ y \]

where \( x \in L \), \( y \in L^* \)

Is \( L^* \) always infinite?

What is \( L^* \) when \( L = \phi \)?

\[ L^* = \Sigma \in \varepsilon \cup \phi \cup \phi \cup \phi \]

What about \( L = \{ \varepsilon \} \)?

\[ L^* = \{ \varepsilon \} \cup \{ \varepsilon \} \cup \{ \varepsilon \} \cup \{ \varepsilon \} \cup \ldots = \varepsilon \in \varepsilon \]

\[ L = \{ \varepsilon \} \Rightarrow L^* \text{ is infinite.} \]
Lemma 2.1. The following identities hold for all languages $A$, $B$, and $C$:

(a) $A \cup B = B \cup A$.
(b) $(A \cup B) \cup C = A \cup (B \cup C)$.
(c) $\emptyset \cdot A = A \cdot \emptyset = \emptyset$.
(d) $\{\varepsilon\} \cdot A = A \cdot \{\varepsilon\} = A$.
(e) $(A \cdot B) \cdot C = A \cdot (B \cdot C)$.
(f) $A \cdot (B \cup C) = (A \cdot B) \cup (A \cdot C)$.
(g) $(A \cup B) \cdot C = (A \cdot C) \cup (B \cdot C)$.

Lemma 2.2. The following identities hold for every language $L$:

(a) $L^* = \{\varepsilon\} \cup L^* = L^* \cdot L^* = (L \cup \{\varepsilon\})^* = (L \setminus \{\varepsilon\})^* = \{\varepsilon\} \cup L \cup (L^+ \cdot L^+)$.
(b) $L^+ = L \cdot L^* = L^* \cdot L = L^+ \cdot L^* = L^* \cdot L^+ = L \cup (L^+ \cdot L^+)$.
(c) $L^+ = L^*$ if and only if $\varepsilon \in L$.

Lemma 2.3 (Arden’s Rule). For any languages $A$, $B$, and $L$ such that $L = A \cdot L \cup B$, we have $A^* \cdot B \subseteq L$. Moreover, if $A$ does not contain the empty string, then $L = A \cdot L \cup B$ if and only if $L = A^* \cdot B$.

**Regular Languages**

$L$ is regular means

- either $L = \emptyset$
- or $L = \varepsilon w \varepsilon$ for some string $w$

if-then-else

- or $L = A \cup B$ for regular $A, B$

Sequence of lines

- or $L = A \cdot B$ for regular $A, B$

Loop

- or $L = A^*$ for regular $A$
REGULAR EXPRESSIONS

0 + 10* = 00* U 01* U 00* U 1* U 1* 0

0, 1, 10, 100, ....

Eg: The language of alternating 0s and 1s.
strings in language
\( \varepsilon, 0101, 0, 1, 10, 101, \ldots \)
strings not in language
11, 0010, 01101

Regular Expression: \((\varepsilon + 1)(01)^* (\varepsilon + 0)\)
A regular expression tree for $a^*a + a^*1(10^*1 + 01^*0)^*10^*$

**Proof:** Let $R$ be an arbitrary regular expression.
Assume that **every regular expression smaller than** $R$ **is perfectly cromulent.**
There are five cases to consider.

- Suppose $R = \emptyset$.

  Therefore, $R$ is perfectly cromulent.

- Suppose $R$ is a single string.

  Therefore, $R$ is perfectly cromulent.

- Suppose $R = S + T$ for some regular expressions $S$ and $T$.
The induction hypothesis implies that $S$ and $T$ are perfectly cromulent.

  Therefore, $R$ is perfectly cromulent.

- Suppose $R = S \cdot T$ for some regular expressions $S$ and $T$.
The induction hypothesis implies that $S$ and $T$ are perfectly cromulent.

  Therefore, $R$ is perfectly cromulent.

- Suppose $R = S^*$ for some regular expression $S$.
The induction hypothesis implies that $S$ is perfectly cromulent.

  Therefore, $R$ is perfectly cromulent.

In all cases, we conclude that $w$ is perfectly cromulent. \(\square\)