Last Time: Naïve Matcher and Rabin Karp Algorithm.
Today: Matching with Finite Automata.

Finite Automaton, \( M = \langle Q, S, f, q_0, F \rangle \)
\( Q = \) finite set of states,
\( S = \) finite set of alphabets,
\( f = \) transition function,
\( q_0 = \) initial state,
\( F = \) final state
\( t = \) text,
\( p = \) pattern.

IDEA: Convert \( p \) into Finite Automaton (FA) and then run \( t \) through FA.

Suffix function: \( \varsigma_p \) (x)

- \( \varsigma_p : \Sigma^* \rightarrow \{0,1,2,\ldots,m\} \)
- Finds the length of the longest prefix of \( p \) that is prefix of \( x \).
  \( \varsigma_p = \{|w| : p=wy, y \in S^*, w \upharpoonright x\} \)

Example:

Let say \( p = ab \)
\( x = \lambda \)
\( \varsigma_p = 0 \)
if \( x = a b c a \)
\( \varsigma_p(a b c a) = 1 \)

Q. How can we construct an automaton \( M \)?
Q. Can we do it fast?

Construction of \( M \) : preprocessing

Let’s say \( p = a b a b c \) and \( t = a b a b c \)

Now, if \( t = a b a b a b a b c a b c \)
\( q = 4 \)
\( a b a b a = t_{[3,7]} \)
\( a b a b c = p \)
So,

\[ \zeta_p(t_{p,7}) \]

\[ q = ? \]

Ans - it means we have found pattern \( p \) up to length \( q \).

So then,

Thus, the transition function,

\[ f(0, abab) = 4 \]

\[ f(4, a) = ? \]

\[ = 3 \]

\[ f(q, a) = \zeta_p(w_q, a) \]

where \( w_q \) is a prefix of \( p \) and \( |w_q| = q \)

The state set \( Q \) is \( \{0, 1, \ldots, m\} \) where \( m = |P| \). The start state \( q_0 \) is state 0, and state \( m \) is the only accepting state.

Table of \( f(q, a) \):

<table>
<thead>
<tr>
<th>( q ) \ ( a )</th>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>(initial state)0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

For the string matching we just loop through 1 to \( n \) and check if \( T[1:i] \) is accepted by the automaton.
Q. is it too costly?

Cost Analysis:

- Search : $O(n)$
- Preprocessing: $O(m^3|S|)$
  Since, locations to be filled = $m + 1 \times |S|$
- Total Cost: $O(n) + O(m^3|S|)$

Kunth-Morris-Pratt (1977)

IDEA: Keep using finite automata for search but avoid constructing $f(q, a)$

Prefix function, $\Pi : \{1, 2, \ldots, m\} \rightarrow \{0, 1, 2, \ldots, m-1\}$

- The longest prefix of $p$ that is a proper suffix of $w_q$

$\Pi_p(q) = \text{maximum } \{|w| : p = wy, w \supseteq p[1,q], |w| < q\}$

Example,

$p = a \ b \ a \ b \ a \ b \ c$

\[\begin{array}{c}
\underbrace{a \ b \ a} \\
\underbrace{a \ b \ a \ b \ c} \\
\end{array}\]  
\[\begin{array}{c}
\Pi_p(1) = 0 \\
\Pi_p(2) = 0 \\
\Pi_p(3) = 1 \\
\Pi_p(4) = 2 \\
\Pi_p(5) = 3 \\
\Pi_p(6) = 4 \\
\Pi_p(7) = 0 \\
\end{array}\]

Continued in next lecture…