Knuth-Morris-Pratt algorithm

It turns out that preprocessing of $P$ can be done in $O(m)$ time, and matching still taking $O(n)$ time. KMP algorithm thus matches strings in $\Theta(m+n)$ time.

Prefix function $\pi$

Given $P[1...q]$ matches text $T[s+1...s+q]$ what is the least $s'>s$ s.t. $P[1...k] = T[s'+1...s'+k]$ for $s'+k = s+q$?
Equivalently, we ask:

What is the largest $K < q$ s.t. $P_K \supseteq P_q$?

Then, $S' = S + (q-k)$ is next potentially valid shift.

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

$$\pi[q] = \max \{k : k < q \mid P_k \supseteq P_q\}.$$  

$\pi[q]$ is the length of the longest prefix of $P$ that is a proper suffix of $P_q$.

$$
\begin{array}{cccccccccccc}
    i & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\
    P[i] & a & b & a & b & a & b & a & b & a & c & a \\
    \pi[i] & 0 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 0 & 1 \\
\end{array}
$$
KMP-Match(T, P)

\[ n := \text{length}(T) \]
\[ m := \text{length}(P) \]
\[ \Pi := \text{Prefix}(P) \]
\[ q := 0 \]

for \( i := 1 \) to \( n \)

while \( q > 0 \) and \( P[q+i] \neq T[i] \)

1. \[ q := \Pi[q] ; \]
   if \( P[q+i] = T[i] \)
   then \( q := q + 1 ; \)
   if \( q = m \)
   then print "match with shift i-m".

2. \[ q := \Pi[q] ; \]

3. \[ q := \Pi[q] ; \]

Time analysis:
- \( q \) is always non-negative.
- It decreases in 1 and 3.
- While loop cannot have complexity more than the decrease in \( q \).
- Total increase in \( q \) is \( O(n) \) in the for loop which are decreased in 1 and 3.
- So, total complexity is \( O(n) \).
Prefix (P)

\[ m := \text{length}[P] \]
\[ \pi[1] := 0 \]
\[ K := 0 \]

for \( q := 2 \) to \( m \)

while \( K > 0 \) and \( P[K+1] \neq P[q] \)

\[ K := \pi[K] ; \]

if \( P[K+1] = P[q] \) then

\[ K := K + 1 ; \]

\[ \pi[q] := K ; \]

return \( \pi \)

Time analysis: Similar as before.

It is \( \mathcal{O}(m) \).

**P:**

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

\( q = 2 \) \( \rightarrow \) \( \pi[2] = 0 \)

\( q = 3 \) \( \rightarrow \) \( K = 1 \), \( \pi[3] = 1 \)

\( q = 4 \) \( \rightarrow \) \( P[3] = P[4] \) \( \rightarrow \) \( K = 2 \) \( \rightarrow \) \( \pi[4] = 2 \)

\( q = 5 \) \( \rightarrow \) \( P[3] = P[5] \) \( \rightarrow \) \( K = 3 \) \( \rightarrow \) \( \pi[5] = 3 \)

\( q = 6 \) \( \rightarrow \) \( P[4] = P[6] \) \( \rightarrow \) \( K = 4 \) \( \rightarrow \) \( \pi[6] = 4 \)
Correctness of the KMP algorithm needs a proof. See the book.