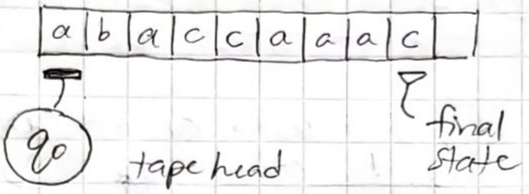
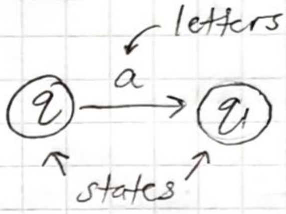
Ch2. Combinatorial Pattern Matching Algorithms

10/14/2024

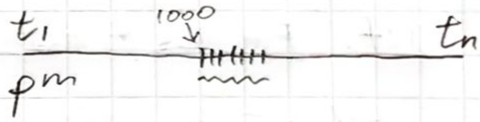
2.1. DFA, NFA, Regular Expressions
2.2. Knuth-Morris-Pratt Algorithm (KMP)

DFA





Pb. <u>Input</u> text $t = t_1 t_2 \ldots t_n$
pattern $p = p_1 p_2 \ldots p_\ell$
$n >> \ell$
$t_i, p_j \in A$

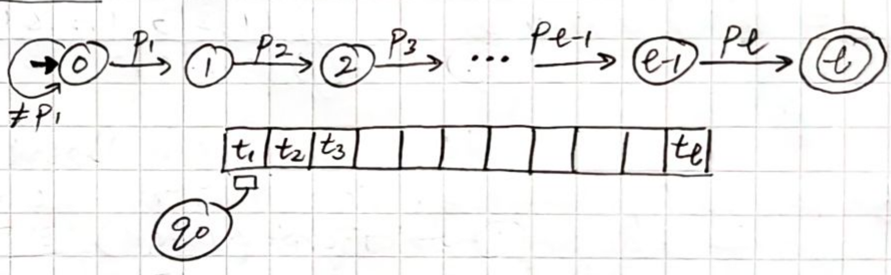


$n = 100{,}000$
$\ell = 150$

<u>Output</u> The position in t of the first occurrence of p

2.2. KMP Algorithm

Skeletal DFA
$\boxed{p = p_1 p_2 \ldots p_\ell}$
DFA construction based only on p
M_p



<u>IF</u> $t_1 = p_1$ <u>THEN</u> M_p enters state 1
The head moves to t_2

<u>IF</u> $t_1 \neq p_1$ <u>THEN</u> M_p stays in state 0
the head moves to t_2

an inductive argument
"base case"

Suppose after having "read" $t_1 t_2 \ldots t_k$ we have M_p in state j. This implies that the last j letters of $\underbrace{t_1 t_2 \ldots t_k}$ are $p_1 p_2 \ldots p_j$.

$p_1 p_2 \ldots p_j$ The suffix of the prefix of length k of t is the prefix of length j of p

Induction hypothesis

"inductive step"

IF $t_{k+1} = p_{j+1}$ THEN M_p enters state $j+1$ and advances head to t_{k+2}

IF $t_{k+1} \neq p_{j+1}$ THEN M_p enters the highest number state i such that $p_1 p_2 \ldots p_i$ is a suffix of $t_1 t_2 \ldots t_k t_{k+1}$

To help with discovering this i, M_p uses an integer valued function f called the Failure Function (*)

Def. Failure Function $f: \{1, 2, 3, \ldots, \ell\}$

$f(j)$ = the largest $\boxed{s < j}$ such that $p_1 p_2 \ldots p_s$ is a suffix of $p_1 p_2 \ldots p_j$

$$p_1 p_2 \ldots p_s = p_{j-s+1} \ldots p_{j-1} p_j$$

otherwise $f(j) = 0$

$p = aabbaab$

j	1	2	3	4	5	6	7
$f(j)$	0	1	0	0	1	2	3

(*) We will present an algorithm for the construction of the Failure Function. (later)

To see how the failure function is used by M_p, let us define $f^{(m)}(j)$ as follows:

i) $f^{(1)}(j) = f(j)$
ii) $f^{(m)}(j) = f(f^{(m-1)}(j))$ for $m > 1$

i.e. $f^{(m)}(j) = \underbrace{f(f(f \ldots f(j)))}_{m}$

$f(6) = 2$, $f(2) = 1 \Rightarrow f^{(2)}(6) = 1$

Suppose again M_p is in state j having read $t_1 t_2 \ldots t_k$ and $t_{k+1} \neq p_{j+1}$. At this point M_p applies the failure function repeatedly to j until it finds the smallest value of m for which either

case 1 $\quad f^{(m)}(j) = u$ and $t_{k+1} = p_{u+1}$ or

case 2 $\quad f^{(m)}(j) = 0$ and $t_{k+1} \neq p_1$

That is, M_p backs up through states $f^{(1)}(j)$, $f(f^{(1)}(j)) = f^{(2)}(j)$, ... until case 1 or case 2 shows up for $f^{(m)}(j)$ but not for $f^{(m-1)}(j)$.

In Case 1, M_p enters state $u+1$
In Case 2, M_p enters state 0
In either case, the head is advanced to t_{k+1}

In Case 1, it is easy to verify that if $p_1 p_2 \dots p_j$ was the longest prefix of p that is a suffix of $t_1 t_2 \dots t_k$, then $p_1 p_2 \dots p_{f^{(u)}(j)+1}$ is the longest prefix of p that is a suffix of $t_1 t_2 \dots t_k t_{k+1}$

In Case 2, no prefix of p is a suffix of $t_1 t_2 \dots t_k t_{k+1}$

M_p then reads t_{k+2}. M_p continues operating in this fashion either until it enters the final state ℓ, in which case we know that the last input symbol of gives us a complete instance of pattern $p = p_1 p_2 \dots p_\ell$, or until M_p has read the last symbol of t without entering final state ℓ

$p =$ a a b b a a b $\qquad A = \{a, b\}$
$t =$ a b a a b a a b b a a b