CSL 356 lecture 35 Oct 31

String matching ... contd

Y : string of length m

X : pattern of length $n < m$

$Y(i) = Y_i Y_{i+1} \cdots Y_{n+i-1}$ (substring of length n starting at i)

$X(i) = X_1 X_2 \cdots X_i$

$A \sqsubset B$: A is a suffix of B

$f(i) = \max_{j<i} X(j) \sqsubset X(i)$

shift function

Eg $X = \underbrace{1\,0\,1\,0\,1}_{X(5)}\,1$

$X(1) \sqsubset X(5)$

$X(3) \sqsubset X(5)$

$\Rightarrow f(5) = 3$

		1	2	3	4	5	6	7	8	
X		a	b	a	b	a	b	c	a	
f(i)		0	0	1	2	3	4	0	1	
Y	\$	a	b	a	b	a	<u>a</u>	\$	\$	\$
				a	b	a	b	a	\$	\$
						a	b	a	b	a

1. Longest prefix of $X(5)$ that matches the suffix of $a\,b\,a\,b\,a\,\underline{a}$ or $X(5).\underline{a}$

2. longest prefix of $X(5)$ that matches a suffix of $X(5) = a\,b\,a = X(3)$

i.e. $f(5) = 3$

Recall that in the potential function based amortised analysis, the amortised work in step i

$$A_i = W_i + \Delta(\phi)$$

actual work

change in potential

Total amortised work

$$= \text{Total work} + \phi_n - \phi_0$$

Potential function for the string matching = extent of partial match

i.e. i, if we have matched upto $X(i)$

Compare X_{i+1} with Y_{j+i} (window starts at Y_j)

Case $X_{i+1} = Y_{j+i}$ Amortised cost

$= 1 + 1 = 2$

actual cost of comparison

Case $X_{i+1} \neq Y_{j+i}$ Amortised cost ≤ 0

We keep track of the total # of comparision w.r.t. to a position in Y.

The amortized cost that can be charged to any position of $Y \leq 2$ (since mismatch only reduces potential)

$\Rightarrow$ Total # comparisons $\leq 2 \cdot m$

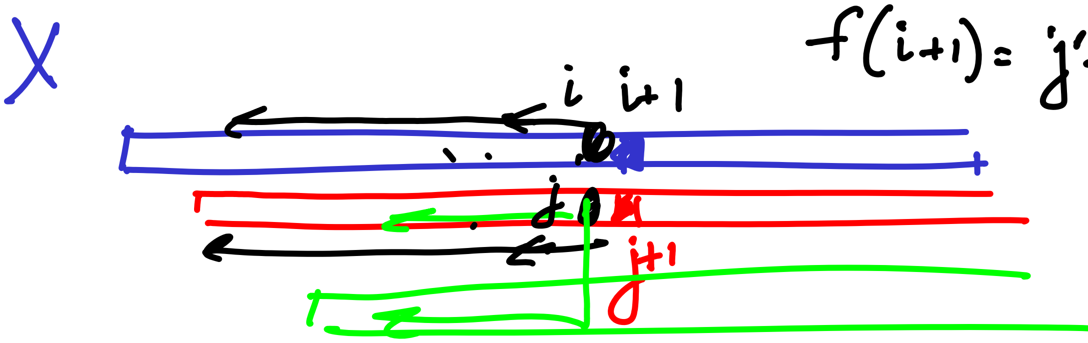
<u>Cost of computing shift function</u>

	1	2	3	4	5	6	7	8
$X =$	a	b	a	b	a	b	c	a
$f(i)$	0	0	1	2	3	4	0	1

Inductively suppose we have computed $f(1)\ f(2) \ldots\ f(i)$. $\boxed{\text{If } f(i) = j \text{ and } X_{i+1} = X_{j+1}}$ then $f(i+1) = j+1$

else if $f(f(i)) = j'$ and $X_{i+1} = X_{j'+1}$

$f(i+1) = j'+1$



It is similar to the string matching algorithm itself. Using a similar analysis, we can bound the running time to $2n$.

So total time for Knuth-Morris-Pratt (KMP) is $O(n+m)$