Algoritmusok és adatszerkezetek 2.

Fekete István előadása alapján
Készítette: Nagy Krisztián

9. előadás

<< KMP mintaillesztés folytatása >>

Lényeg:

M előzetes elemzésével $\Longrightarrow$ next függvény nagyobb lépésben tud haladni

Példa:
M: $A\ B\ A\ B\ A\ C$
$next$: $0\ 0\ 1\ 2\ 3$ –

$next[j]$ a j-vel lezárólag fedésben levő mintában a leghosszabb prefix-suffix pár mérete.

Alkalmazás példa:
S: $A\ B\ A\ B\ A\ B\ A\ C$
M: $A\ B\ A\ B\ A\ C$

$next[5] = 3 \Longrightarrow$
S: $A\ B\ A\ B\ A\ B\ A\ C$
M: $A\ B\ A\ B\ A\ C$
Nem kell ellenőrizni!

$Initnext$ → next függvény értékének előzetes kiszámítása

Naív megoldás:

3-szoros ciklussal

1. ciklus: $j = 1 \dots m - 1$
2. ciklus: $h = 1 \dots j - 1$
3. ciklus: $Lineáriskeresés2(h)$

Műveletigénye: $T(m) = O(m^3)$. Ez pedig nem hatékony!

Ötlet: M eltolása önmagán KMP-elv szerint a next már kiszámított értékei alapján.

Példa:
M: $A\ B\ A\ B\ A\ C\ A\ B\ D$
$A\ B\ A\ B\ A\ C\ A\ B\ D$ külön: $next[1] = 0$ mindig
$i = 1\ j = 0$ $next(2) = 0$
$i = 2\ j = 0$ $next(3) = 1$
$i = 3\ j = 1$ $next(4) = 2$

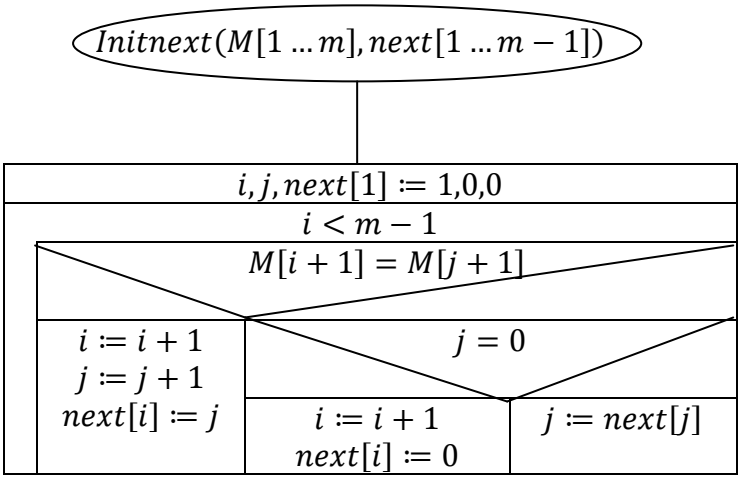
$i = 4\ j = 3\ \ next(5) = 3$

M: $A\ B\ A\ B\ A\ C\ A\ B\ D$
$A\ B\ A\ B\ A\ C\ A\ B\ D$

$i = 5\ \ j = 3\ \ \rightarrow elromlik!\ \ next(3) = 1$ alapján 2 lépéssel tudjuk csúsztatni a mintát.
Az első A-t a 3. betűre (,ami szintén A) csúsztatjuk

$A\ B\ A\ B$ …
elromlik! $next(6) = 0$

$i = 7\ j = 0\ \ next(7) = 1$
$i = 8\ j = 1\ \ next(8) = 2$ KÉSZ



$T(n,m) = \theta(n+m)$ bizonyítás nélkül!

33. Gyorskeresés (Quick search) [keresés->mintaillesztés]

Elve: Más, mint a KMP-nél.
A minta után pozíció alapján (karakter alapján) történik a léptetés.

Példa:

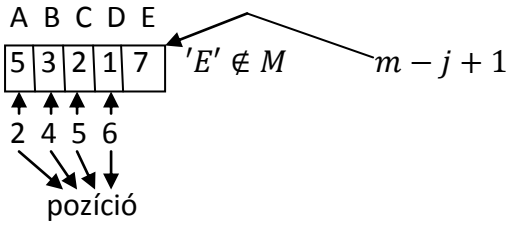
$\hat{k}$
S: … $A\ B\ A\ A\ C\ B\ \boxed{C}\ D$ $\quad S[k+m+1]$
M: $A\ A\ \boxed{C}\ B\ \boxed{C}\ D$
$\hat{j}..\hat{j}$
$\longrightarrow$
léptetek

$1 \dots m$ léptetés esetén mindig kerül egy betű az $S[k+m+1]$ alá.
Szeretnénk a C alá C-t tolni. Jobbról az első C-t toljuk alá (Mintában hátulról / visszafelé)
Ha $C \notin M$ lenne, akkor $S[k+m+2]$ alá tolnánk a minta elejét.

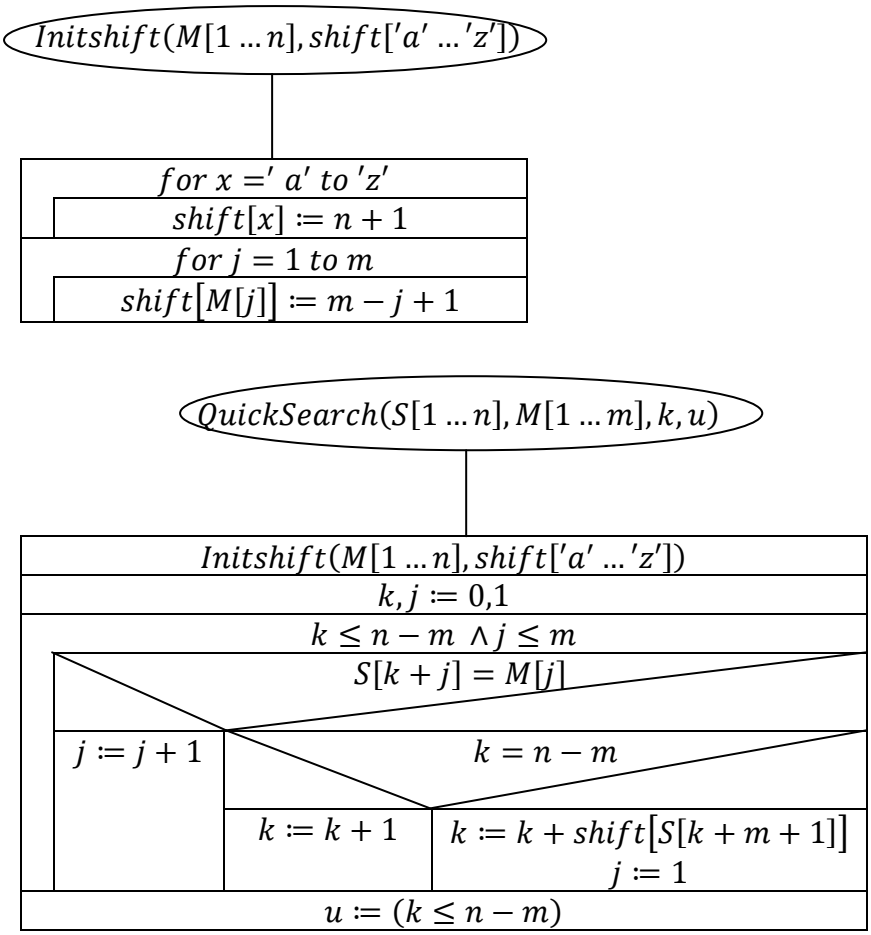
Jobbról az első 'C' helyét a shift függvény adja meg. Ezt előzetesen kiszámítjuk.

$shift['a' \dots 'z'] \rightarrow$ 'a' … 'z' ahány betű van az ábécében.

Például:
$shift[1 \dots 5]$:



A ~~7~~ ~~6~~ 5
B ~~7~~ 3
C ~~7~~ ~~4~~ 2
D ~~7~~ 1
E 7



Műveletigény:

$mO(n,m) = \theta\left(\frac{n}{m+1}\right)$

$MO(n,m) = \theta(nm)$