Algoritmusok és adatszerkezetek 2.

Fekete István előadása alapján
Készítette: Nagy Krisztián

10. előadás

34. Rabin-Karp mintaillesztés

Elv: Strukturális. Itt Més S-nek az $|M|$ hosszú szeleteit egész számokra képezzük egy hasító függvénnyel és ezekkel dolgozunk tovább. Általában ezek nagy számok, ezért $mod\ p$ aritmetikát alkalmazunk, ahol p egy elég nagy szabadon választható prím.

$x = y$ ⇒ ugyanahoz a maradékosztályhoz tartoznak

Példa:
Legyen $x = 1102$ a minta kódja
Legyen most az ábécénk 10 betűs:

A	B	C	D	E	F	G	H	I	J
0	1	2	3	4	5	6	7	8	9

$M: BBAC$
$S:\ DACABBAC$
$S_1: 3020$
$S_2:\ \ 0201$
$S_3:\ \ \ 2011$
$S_4:\ \ \ \ 0110$
$S_5:\ \ \ \ \ 1102$ ⇒ egyezik a minta kódjával

Amennyiben az összehasonlítás nem egyezik meg, úgy tolást hajtunk végre. Ez az alapelv.

Általános jelölések:

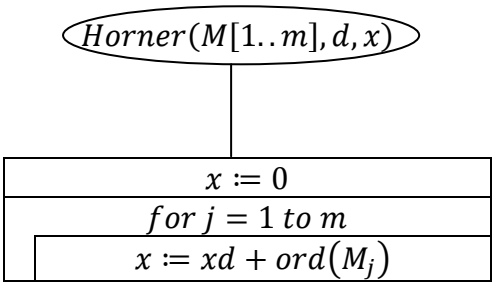
Ábécé: $H \coloneqq \{A, B, C, \dots, Z\}$
Ábécé számossága: $d \coloneqq |H|$
Függvény, ami megadja egy betű ábécébeli sorszámát: $ord: H \to [0 \dots d-1]$

$$x = \sum_{j=1}^{m} ord(M_j) d^{m-j}$$

Ezt az x-et a Horner-algoritmus segítségével hatékonyabban ki tudjuk számolni.

$$x = \sum_{i=1}^{n} a_i d^{n-i} = a_1 d^{n-1} + a_2 d^{n-2} + \cdots + a_{n-1} d + a_n$$

$x \coloneqq 0$
$x \coloneqq xd + a_1 \Rightarrow x = a_1$
$x \coloneqq xd + a_2 \Rightarrow x = a_1 d + a_2$
$x \coloneqq xd + a_3 \Rightarrow x = a_1 d^2 + a_2 d + a_3$ …



A fentebbiek ismeretében ki tudjuk számolni x-et és s_1-et.
s_{i+1} számolható s_i-ből.

Például:

$s_1 \coloneqq 3020$
$s_2 = (3020 - 3 \cdot 10^3) \cdot 10 + 1 = 0201$
$\boxed{s_{i+1} = [(s_i - ord(s_i) \cdot d^{m-1})] \cdot d + ord(s_{m+i})}$

Az s_i számok „nagyok” $\Leftrightarrow$ túlcsordulás következik be a program működése során

Csak pozitív (nem-negatív) számokra nézve 4 bájt maximum értéke: $2^{32} - 1 \approx 2^{32}$
Legyen az ábécénk számossága $d = 32$ továbbá a mintánk hossza $|M| = 8$, ekkor $d^{m-1} = 32^7 = (2^5)^7 = 2^{35}$. Láthatjuk, hogy ez nagyobb nagyságrendű, mint a 2^{32}.

A kiküszöböléshez modulo-aritmetikát vezetünk be.

Válasszunk olyan nagy p prímet, amelyre $p \cdot d$ még ábrázolható.

Az egyenlőség vizsgálata $x =^{?} s_i$ helyett az $x\ (mod\ p) =^{?} s_i\ (mod\ p)$-kérdést vizsgáljuk.
Kérdés: Ugyan ahhoz a maradékosztályhoz tartozik-e x *és* s_i ?

Két lehetséges válasz van a fentebbi kérdésre:

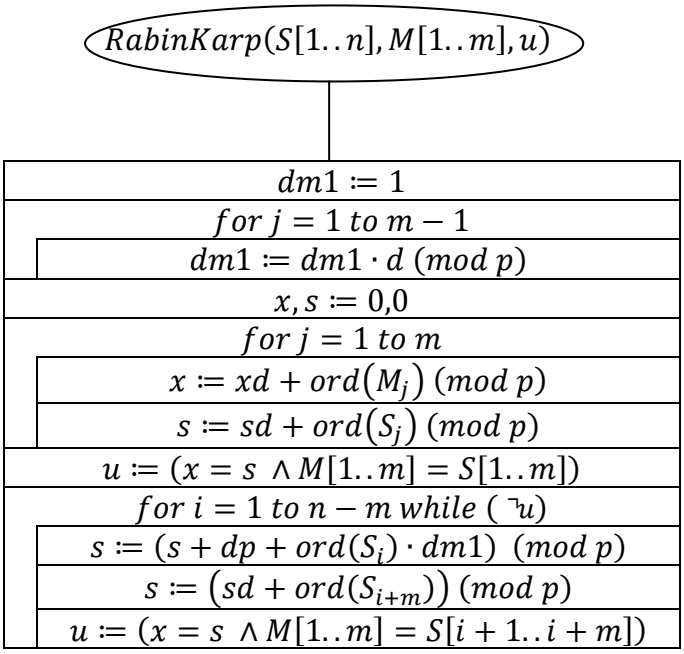
1. $x\ (mod\ p) \neq s_i\ (mod\ p) \Rightarrow x \neq s_i$
2. $x\ (mod\ p) = s_i \Rightarrow ?$
   Ekkor x *és* s_i számjegyeit mint karakterpárokat, karakterenként összehasonlítjuk.
   Nagy p prím esetén az látható, hogy a hamis találat ritka lesz.

Tehát a formula:
$s_{i+1} \coloneqq \left([(s_i - ord(s_i) \cdot d^{m-1})] \cdot d + ord(s_{m+i})\right)\ (mod\ p)$

A modulo számítása közben újabb probléma merült fel: $(s_i - ord(s_i) \cdot d^{m-1})$ értéke lehet negatív is. Ezt a problémát a $d \cdot p$ hozzáadásával korrigáljuk (Ez biztosítani fogja, hogy a kifejezés ne legyen negatív, tobvábbá az osztási maradékot se változtatja meg).

$s_{i+1} \coloneqq \left([(s_i + dp - ord(s_i) \cdot d^{m-1})] \cdot d + ord(s_{m+i})\right)\ (mod\ p)$

A kifejezés túlcsordulásának elkerülése végett számítsuk a fentebbi kifejezést két lépésben (modulo esetén megtehetjük):
1. $s = \ (s_i + dp - ord(s_i) \cdot d^{m-1})\ \ (mod\ p)$
2. $s_{i+1} = [s \cdot d + ord(s_{m+i})]\ (mod\ p)$



Megjegyzések:
$dm1$ a $d^{m-1}\ (mod\ p)$-t takarja.
Tegyüg fel, hogy az ∧ művelet az u logikai változóknál egy rövidre zárás a második feltételt csak akkor értékeli ki a program, ha az első feltétel igaz.

Műveletigény:

Legjobb eset: $m\ddot{O}(n,m) = \theta(n)$

Legrosszabb esetben:

- Ha p olyan, hogy mindig hamis találatot kapunk, akkor minden esetben karakterenként is vizsgálnunk kell, ekkor a Brute-force algoritmust kapjuk vissza:

$M\ddot{O}(n,m) = \theta(n \cdot m)$

- Jó p esetén, nincs vagy csak nagyon kevés a hamis találat: $M\ddot{O}(n,m) = \theta(n)$. Tehát egy stabil algoritmus.