Algoritmusok és adatszerkezetek II
Mintaillesztés témakör jegyzete

Készült Ásványi Tibor előadásai és gyakorlatai alapján

Sárközi Gergő, 2021-22-1. félév
Nincsen lektorálva!

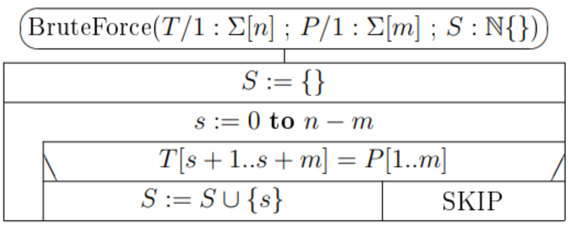
Tartalomjegyzék

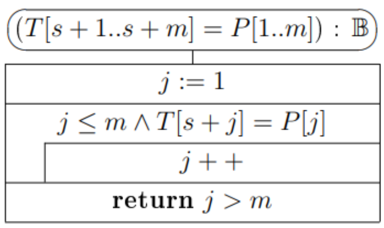
1. Mintaillesztés

- Abécé: $\Sigma = \{\sigma_1, \sigma_2, ..., \sigma_d\}$ $\quad (1 \leq d < \infty$ konstans)
- Szöveg, amiben keresünk: $T/1 : \Sigma[n] \quad (1 \leq n)$
- Minta, amit keresünk: $P/1 : \Sigma[m] \quad (1 \leq m \leq n)$
- $s \in 0..(n-m)$ P érvényes eltolása T-n $\Leftrightarrow T[s+1..s+m] = P[1..m]$
- A cél az érvényes eltolások S halmazának megállapítása

2. Egyszerű (brute force) algoritmus

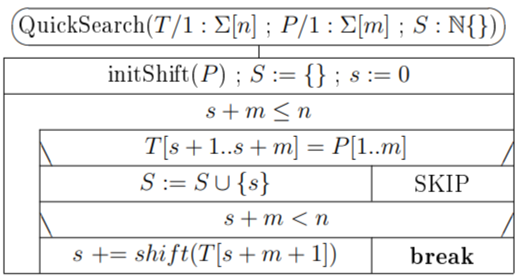
- Minden lehetséges s értékre, egymástól függetlenül, próbáljuk a mintát
- Időkomplexitás: $MT(n,m) \in \Theta(n*m)$ és $mT(n,m) \in \Theta(n)$
  - Alapból $MT \in \Theta((n-m+1)*m)$ és $mT \in \Theta(n-m+1)$
  - $m \leq n \implies (n-m+1) \in \Theta(n)$
  - Tehát $MT \in \Theta(n*m)$ és $mT \in \Theta(n)$ $\quad$ (mint legfelül)
  - Ha m nem elhanyagolható n-hez képest ($m \geq \epsilon * n$ ahol $0 < \epsilon < 1$) akkor $(n*m) \in \Theta(n^2) \implies MT \in \Theta(n^2)$

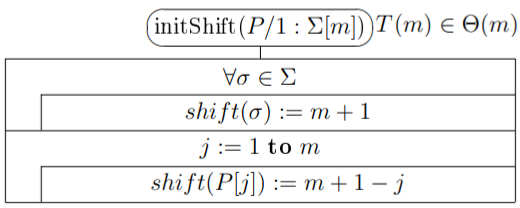




3. Quicksearch

- Egynél nagyobb lépésekben növeli az s eltolását (de nem ugrik át egy érvényes eltolást sem)
- Előkészítő fázis: Ábécé minden σ eleméhez $\mathit{shift}(\sigma) \in 1..m + 1$ címke
  - Csak a mintától függ, a szövegtől nem
- $\mathit{shift}(\sigma)$ működése:
  - σ mindig a minta utáni első karakter a szövegben: $\sigma = T[s+m+1]$
  - Megmondja $T[s + 1..s + m]$ megnézése után mennyivel nőjön s
  - Ha $\sigma \in P$: s mennyivel nőjön, hogy a minta illeszkedhessen a $T[s + m + 1]$ karakterre (pl. ha $P[m] = \sigma$ akkor $\mathit{shift}(\sigma) = 1$)
  - Ha $\sigma \notin P$: minta átugorja $T[s+m+1]$ karaktert ($\mathit{shift}(\sigma) = m+1$)
- Időkomplexitás:
  - $mT \in \Theta(\frac{n}{m+1} + m)$ (pl. T és P diszjunktak)
    * Jobb, mint a brute force megoldás
  - $MT \in \Theta((n - m + 2) * m)$ (pl. T és P mind azonos σ sokszor)
    * Azonos brute force-szal, de gyakorlatban lassabb
  - Átlagosan gyorsabb, mint a brute force, de azért nem optimális





3.1. Quicksearch példa

- Bal fenti ábra:
  - *xxxx* jelöli a mintával az eltolás előtt összehasonlított szövegrészt
  - Az eltolás mértékét mutatja be: az eltolás utána állapot látható

```
Szöveg: ...xxxxA......xxxxB......xxxxC......xxxxD...
 Minta:     CADA            CADA       CADA       CADA
```

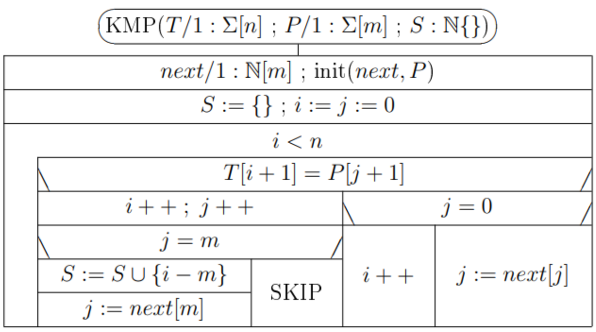
σ	A	B	C	D
$shift(\sigma)$	1	5	4	2

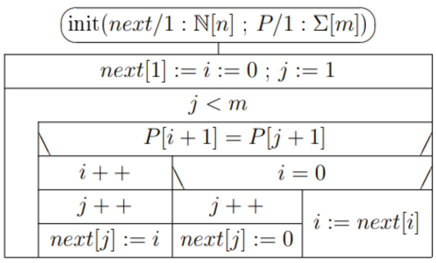
σ	A	B	C	D	
initial $shift(\sigma)$	5	5	5	5	5
C			4		4
A	3				3
D				2	2
A	1				1
final $shift(\sigma)$	1	5	4	2	

$i =$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
$T[i]=$	A	D	A	B	A	B	C	A	D	A	B	C	A	B	A	D	A	C	A	D	A	D	A
	$\cancel{C}$	A	D	A																			
		$\cancel{C}$	A	D	A																		
$s = 6$							$\underline{C}$	$\underline{A}$	$\underline{D}$	$\underline{A}$													
												$\underline{C}$	$\underline{A}$	$\cancel{D}$	A								
														$\cancel{C}$	A	D	A						
$s = 17$																		$\underline{C}$	$\underline{A}$	$\underline{D}$	$\underline{A}$		
																				$\cancel{C}$	A	D	A

4. Knuth-Morris-Pratt (lineáris) algoritmus RÖVIDEN

- Lineáris időben végzi el a feladatot
- Nem kell minden esetben a minta elejétől kezdeni az illesztést:
  a prefixet nem kell újra vizsgálni, ha az egyezik a szufixszel
- Előfeldogozás: ($\Theta(m)$ idő alatt végbemegy)
  - megadunk egy *next* függvényt, ami megadja a leghosszabb megegyező prefix-szuffix párok hosszát minden minta kezdőszeletre (hosszra)
  - $next(j)$ a leghosszabb olyan P prefix hossza, amely P első j karakterének szuffixe (de nem egyezik meg vele), azaz $next(j) \in 0..(j-1)$
- A szövegben nem kell visszaugrani, azaz buffer nélkül is használható. (Minden karaktert csak egyszer olvasunk ki, és csak "előrefelé" haladunk.)
- A mintát sikeres/sikertelen illesztés esetén annyival toljuk előrebb, amerkkora a sikeresen illesztet részminta hossza MÍNUSZ a sikeresen illesztet részminta legnagyobb szuffixe, ami egyben prefix.
  Azaz ez a legnagyobb szuffix lesz a minta kezdete.
- Időkomplexitás: $MT = mT \in \Theta(n)$
  - $\Omega(n)$, mert i egyesével nő és n-ig megy
  - $O(n)$, $2i - j$ értéke mindig szig. mon. nő, tehát max $2n$ iteráció





- $next[1] = 0$
- $next[i+1] \leq next[i] + 1$
- $next(j) \in 0..(j-1) \quad (j \in 1..m)$

- *init* ciklusának invariánsa:
  - $i \le j \le m$
  - P első i karaktere szuffixe P első j karakterének
  - és $\forall l \in (i+2)..j$: P első l karaktere nem szuffixe P első $j+1$ karakterének, de egyenlőek lehetnek
  - és $next[1..j] = next(1..j)$ (azaz a tömb a fv alapján van töltve)
- *KMP* ciklusának invariánsa:
  - $i \in 0..n$ és $j \in 0..(m-1)$ és $j \le i$
  - és $S = \{s \in 0..(i-m) \mid T[(s+1)..(s+m)] = P\}$
  - és P első j karaktere szuffixe T első i karakterének (vagy egyenlőek)
  - és $\forall l \in (j+2)..m$: P első l karaktere nem szuffixe T első $i+1$ karakterének (és nem is egyenlőek)

4.1. Példa

i	j	$next[j]$	1 A	2 B	3 A	4 B	5 B	6 A	7 B	8 A
0	1	0		$\cancel{A}$						
0	2	0			$\underline{A}$					
1	3	1			A	$\underline{B}$				
2	4	2			A	B	$\cancel{A}$			
0	4	2					$\cancel{A}$			
0	5	0						$\underline{A}$		
1	6	1						A	$\underline{B}$	
2	7	2						A	B	$\underline{A}$
3	8	3								

Minta:
P = ABABBABA

A végeredmény:

$P[j] =$	A	B	A	B	B	A	B	A
$j =$	1	2	3	4	5	6	7	8
$next[j] =$	0	0	1	2	0	1	2	3

$i =$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
$T[i]=$	A	B	A	B	A	B	B	A	B	A	B	B	A	B	A	B	A
	$\underline{A}$	$\underline{B}$	$\underline{A}$	$\underline{B}$	$\cancel{B}$												
s=2			A	B	$\underline{A}$	$\underline{B}$	$\underline{B}$	$\underline{A}$	$\underline{B}$	$\underline{A}$							
s=7								A	B	A	$\underline{B}$	$\underline{B}$	$\underline{A}$	$\underline{B}$	$\underline{A}$		
													A	B	A	$\underline{B}$	$\cancel{B}$
															A	B	$\underline{A}$

$$S = \{\,2; 7\,\}$$

5. Knuth-Morris-Pratt (lineáris) algoritmus

5.1. Jelölések

- Akár teljes prefix: $x \sqsubseteq y \Leftrightarrow \exists z : x + z = y$
- Igazi prefix: $x \sqsubset y \Leftrightarrow x \sqsubseteq y \wedge x \neq y$
- Akár teljes szuffix: $x \sqsupseteq y \Leftrightarrow \exists z : z + x = y$
- Igazi szuffix: $x \sqsupset y \Leftrightarrow x \sqsupseteq y \wedge x \neq y$
- Az üres sztring mindennek a prefixe és a szuffixe is.
- Kezdőszelet: $A_j = A[1..j]$ (ezt a jelölést ritkán használjuk)
  - A_0 az üres sztring
- Prefix-szuffix: $x \square y \leftrightarrow x \sqsubset y \wedge x \sqsupset y$
- i. legnagyobb elem: $\max_i H \quad (i \in 1..|H|)$
  - $\max_1 H = \max H$ és $\max_{|H|} H = \min H$
- $H(j) = \{h \in 0..j-1 \mid P_h \sqsupset P_j\} = \{|x| \mid x \square P_j\} \quad (j \in 1..m)$
  Azaz azon sztring hosszak, amelyek prefixek és szuffixek is P-nek egyben.
- $next(j) = \max H(j) \quad (j \in 1..m)$
  Leghosszabb P-beli prefix hossza, ami egyben valódi szuffixe P_j-nek.

NINCS BEFEJEZVE, NAGYON HIÁNYOS