Rendezés lineáris időben

Számjegypozíciós rendezés

radix_sort(A : dDigitNumber⟨⟩ ; d : $\mathbb{N}$)

$i := 1$ to d
use a stable sort to sort list A on digit i

$d \in \Theta(1)$

stable_sort(n): $\Theta(n)$ időben rendez

radix sort(n) $\Theta(n)$ időt igényel

	1.		2.	
02	→	30	→	01
11		10		02
30		11		10
10		01		11
01		02		12
12		12		30

rendezett

Stabil segédrendezés: pl. Szétválogató rendezés

The sorting problem: Given abstract list L of length n with element type $\mathcal{T}$, $r \in O(n)$ positive integer, (r: radix)
$\varphi : \mathcal{T} \to 0..(r-1)$ key selection function.

Let us sort list L with stable sort, with linear time complexity.

$n := |L|$

$T(n, r)$

distributing_sort$(L : \mathcal{T}\langle\rangle \,;\, r : \mathbb{N} \,;\, \varphi : \mathcal{T} \to 0..(r-1))$

- $B : \mathcal{T}\langle\rangle[r]$ // array of lists, i.e. bins
- $k := 0$ to $r-1$
  - Let $B[k]$ be empty list // init the array of bins
- L is not empty
  - Remove the first element x of list L } $\Theta(1)$!
  - Insert x at the end of $B[\varphi(x)]$ // stable distribution } $\Theta(1)$!
- $k := r-1$ downto 0
  - $\underbrace{L := B[k] + L}_{\Theta(1)\,!}$ // append $B[k]$ before L

$\Theta(r)$

$\Theta(n)$

$\Theta(r)$

$\Theta(n+r)$

$|K| = r \in O(n)$

$\Theta(n)$

$T(n) \in \Theta(n)$

$S(r) \in \Theta(r) \subseteq O(n)$

02
11
30
10
01
12

szétválogat

r = 4

B_0: 30, 10
B_1: 11, 01
B_2: 02, 12
B_3:

összefűz

30
10
11
01
02
13

szét-válogat

B_0: 01, 02
B_1: 10, 11, 13
B_2:
B_3: 30

összefűz

01
02
10
11
13
30

rendezve: ✓

bins
(polcok v „edények")

C2L-ekre:

radix_sort(L : E2* ; d, r : $\mathbb{N}$)

- $BinHead$: E2[r] // the headers of the lists representing the bins
- B : E2*[r] // pointers to the headers
- $i := 0$ to $r-1$
  - $B[i] := \&BinHead[i]$ // Initialize the ith pointer.
- $i := 1$ to d
  - distribute(L, i, B) // Distribute L on the ith digits of keys.
  - gather(B, L) // Gather form the bins back into L

distribute(L : E2* ; i : $\mathbb{N}$; B : E2*[r])

- $L \rightarrow next \neq L$
  - $p := L \rightarrow next$; unlink(p)
  - precede($p, B[\text{digit}(i, r, p \rightarrow key)]$

gather($B : \text{E2*}[r]$; $L : \text{E2*}$)

$i := 0$ to $r-1$

append$(L, B[i])$ // add to the end of L the elements form $B[i]$

append($L, Bi : \text{E2*}$)

$Bi \rightarrow next \neq Bi$	
$p := L \rightarrow prev$; $q := Bi \rightarrow next$; $r := Bi \rightarrow prev$	SKIP
$p \rightarrow next := q$; $q \rightarrow prev := p$	
$r \rightarrow next := L$; $L \rightarrow prev := r$	
$Bi \rightarrow next := Bi$; $Bi \rightarrow prev := Bi$	

<u>Tömbökre:</u> A radix rendezés segédrendezése a leszámláló rendezés

The sorting problem: Given array $A:\mathcal{T}[n]$, $r \in O(n)$ positive integer, $\varphi : \mathcal{T} \to 0..(r-1)$ key selection function.

Let us sort array A with stable sort and linear time complexity so that the result is produced in array B.

leszámláló rendezés

counting_sort($A, B : \mathcal{T}[n]$; $r : \mathbb{N}$; $\varphi : \mathcal{T} \to 0..(r-1)$)

$T(n,r) \in \Theta(n+n) = \Theta(n)$

$C : \mathbb{N}[r]$ // counter array
$k := 0$ to $r-1$
$C[k] := 0$ // init the counter array
$i := 0$ to $n-1$
$C[\varphi(A[i])]{+}{+}$ // count the items with the given key
$k := 1$ to $r-1$
$C[k] \mathrel{+}= C[k-1]$ // $C[k]$:= the number of items with key $\leq k$
$i := n-1$ downto 0
$k := \varphi(A[i])$ // k := the key of $A[i]$
$C[k]--$ // The next one with key k must be put before $A[i]$ where
$B[C[k]] := A[i]$ //Let $A[i]$ be the last of the {unprocessed items with key k}

$\Theta(r)$

$\forall k \in [0..r): C[k] = 0$

$\Theta(n)$

$\forall k \in [0..r): C[k] = |\{i \in [0..n) \mid \varphi(A[i]) = k\}|$

$\Theta(r)$

$\sim$ $\varphi(A[i]) \leq$ $\leq k$

$\Theta(n)$

$S(n,r) \in \Theta(n+r) = \Theta(n)$ ha $r \in O(n)$

radix_sort($A : \mathbb{N}[n]$; $r, d : \mathbb{N}_+$) $2 \mid d$

digit($i, r : \mathbb{N}_+$; $x : \mathbb{N}$) : $\mathbb{N}$
adott: ✓

$B : \mathbb{N}[n]$; $i := 1$
$i \leq d$
counting_sort(A, B, r, digit($i, r, .$))
i++
counting_sort(B, A, r, digit($i, r, .$))
i++

$d \in \Theta(1)$
$\Downarrow$
$T(n) \in \Theta(n)$

A

02
11
30
10
01
12

	C	02	11	30	10	01	12	Σ	12	01	10	30	11	02
0	0			1	2			2			1	0		
1	0		1			2		4		3			2	
2	0	1					2	6	5					4
3	0							6						

B

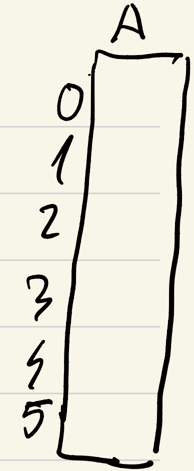
0	30
1	10
2	11
3	01
4	02
5	12

B

0	30
1	10
2	11
3	01
4	02
5	12

HF

counting sort (B, A, ...)
szemléltetése
a jobbról 2. szj. szerint.



$r = 256$

$s := r-1 = \underbrace{1 \ldots 1}_{8}{}_{2}$

$digit(i, r, x) \sim x >> ((i-1)*8) \,\&\, s$

Lehet egy segédvált.-ban.

$\forall$ kulcs $\in [0;1)$ A kulcsok egyenletesen oszlanak el az intervallumon.

bucket_sort(L : list)

idő

- n := the length of L
- B : list[n] // Create the buckets $B[0..(n-1)]$
- $j := 0$ to $(n-1)$
  - Let $B[j]$ be empty list

} $\Theta(n)$

- $L \neq \varnothing$
  - Remove the first element of list L
  - Insert this element according to its key k into list $B[\lfloor n * k \rfloor]$

} $\Theta(n)$

min: $\Theta(n)$

- $j := 0$ to $(n-1)$
  - Sort list $B[j]$ nondecreasingly

} Átl.: $\Theta(n)$ idő

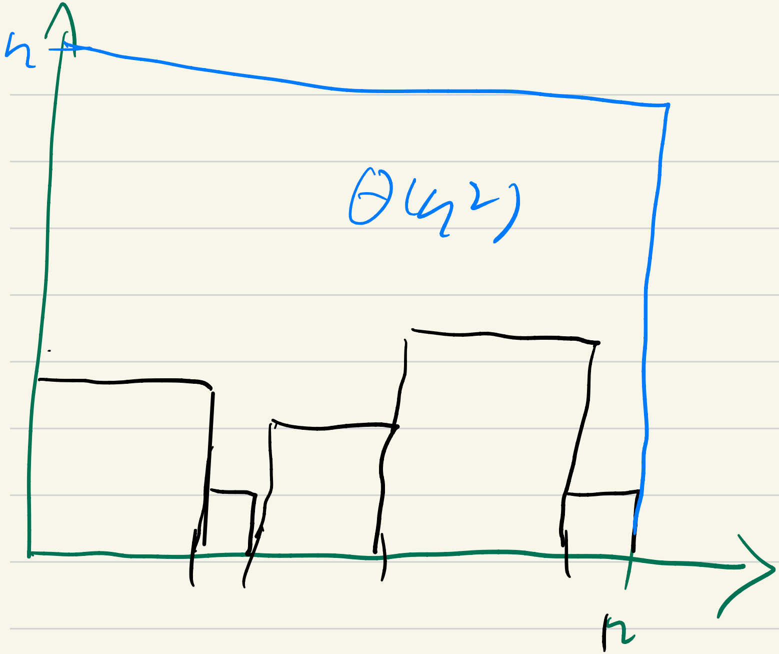
- Append lists $B[0], B[1], \ldots, B[n-1]$ in order into list L

} $\Theta(n)$

$k \in [0;1) \Rightarrow n*k \in [0;n) \Rightarrow \lfloor n*k \rfloor \in [0..n)$

$\left.\begin{array}{l} mT(n) \\ AT(n) \end{array}\right\} \Theta(n)$

$S(n) \in \Theta(n)$



$\underline{\text{ha}}$ sort = IS $\Rightarrow$

$\Rightarrow MT(n) \in \Theta(n^2)$

$\underline{\text{ha}}$ sort = MS $\Rightarrow$

$\Rightarrow MT(n) \in \Theta(n \log n)$

Rendezési módszerek összehasonlítása (T, S)

	mT(n)	AT(n)	MT(n)	mS(n)	AS(n)	MS(n)	Stabil
IS	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(1)$	$\Theta(1)$	$\Theta(1)$	+
MS(A)	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	+
MS(L)	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(\log n)$	$\Theta(\log n)$	$\Theta(\log n)$	+
QS(A)	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(\log n)$	$\Theta(\log n)$	$\Theta(n)$	−
HS(A)	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(1)$	$\Theta(1)$	$\Theta(1)$	−
QS(L)	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(\log n)$	$\Theta(\log n)$	$\Theta(n)$	(+)
Q(A)+HS(A)	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(\log n)$	$\Theta(\log n)$	$\Theta(\log n)$	−
DS(L)/RS(L)	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$S(r) \in \Theta(r)$	$S(r) \in \Theta(r)$	$S(r) \in \Theta(r)$	+
CS(A)/RS(A)	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	+
BS(L)+IS(L)	$\Theta(n)$	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	(+)
BS(L)+MS(L)	$\Theta(n)$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	(+)

IS: Insertion Sort / MS: MergeSort / QS: Quicksort / HS: Heapsort
DS: Distributing~ / RS: Radix~ / CS: Counting~ / BS: Bucket~
A: array / L: Linked list / (+): megoldható