

مرتب سازی در زمان خطی - مرتب سازی خطی

آیا می توان الگوریتمی از مرتبه $\Theta(n)$ در برداشتن حالت برای مرتب سازی آرایه ای به طول n داشت ؟ نه - مگر اینکه مرتب سازی بسته برعکس بباشد

~~if~~ \leftarrow \rightarrow

قصص . حداقل زمان اجرا در برداشتن حالت

هر الگوریتم مرتب سازی بسته

معابد از مرتبه $\Omega(n \lg n)$

Comparison-based
sorting algorithm

$$= \Omega(n \lg n)$$

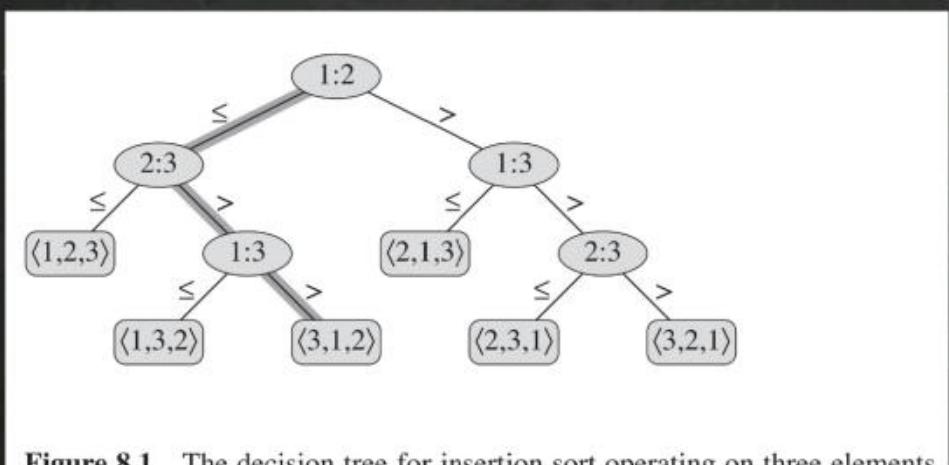


Figure 8.1 The decision tree for insertion sort operating on three elements.

مرتب سازی در زمان خطی - مرتب سازی خطی

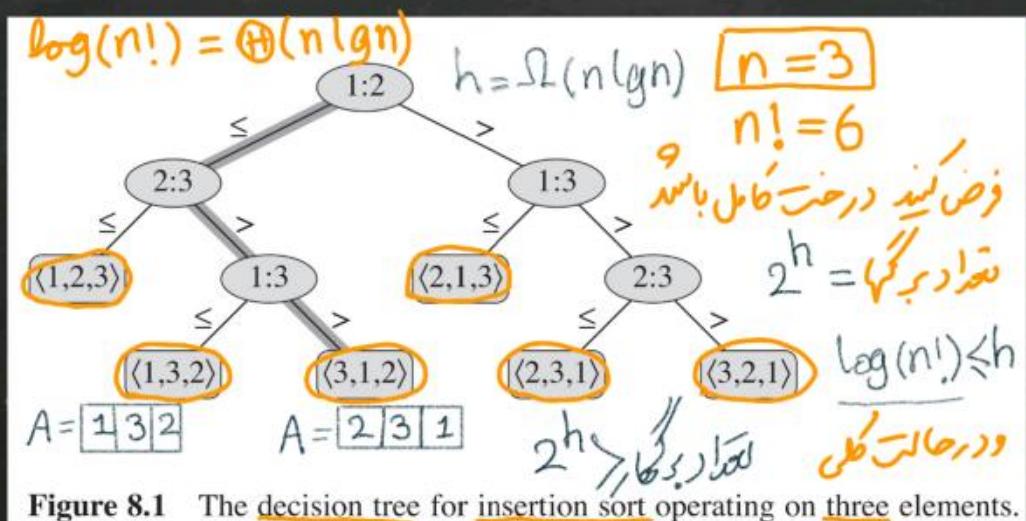


Figure 8.1 The decision tree for insertion sort operating on three elements.

input: $A \downarrow$



$$A = [2, 4, 1, 3, 5]$$

$$A = [9, 15, 7, 12, 20]$$

→ بیان قصنه فون مرتب سازی از موبه $O(n)$ عذر ممکن است.

- ملر آنده الگوریتم را نه باشیم که از معارف اسما $O(n^2)$ نباشد.
- باید است نیازمند مغروضات و اطلاعات و ویژه ای نباید به معابر هستم.

Counting Sort

مرتب سازی سهارسی

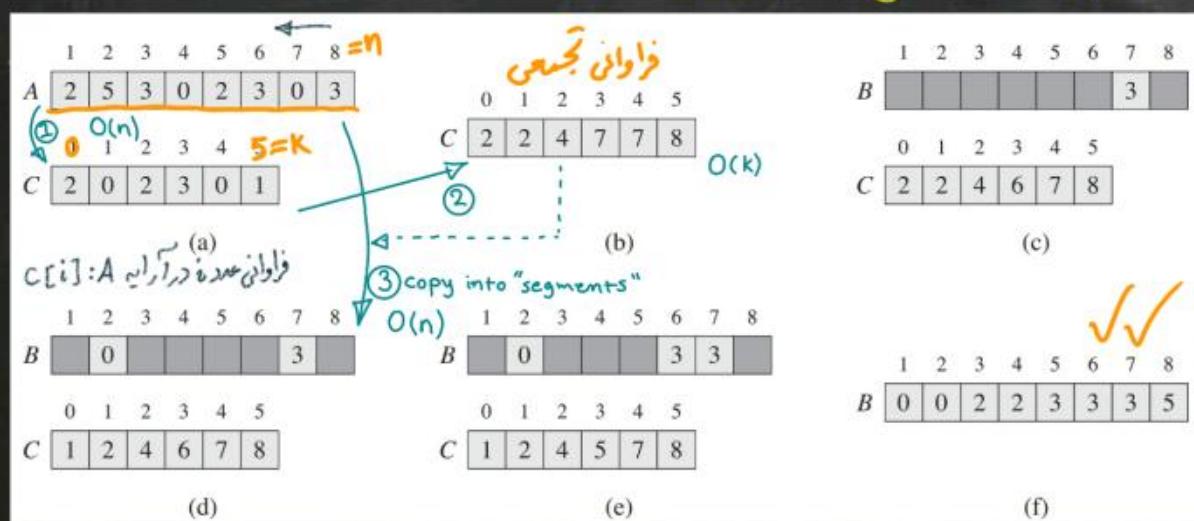
Radix Sort

مرتب سازی مبنای

Bucket Sort

مرتب سازی جعبه ای

- مرتب سازی سهارسی



1	2	3	4	5	6	7	8
0	1	2	3	4	5	6	7

0	1	2	3	4	5
2	2	4	6	7	8

(c)

1	2	3	4	5	6	7	8
0	0	2	2	3	3	3	5

(f)

فرض: معادر
داده سرمه ازین

اعداد صحیح
 $\{0, 1, 2, \dots, k\}$

متن.

COUNTING-SORT(A, B, k)

```
1 let  $C[0..k]$  be a new array
2 for  $i = 0$  to  $k$ 
3    $C[i] = 0$ 
4 for  $j = 1$  to  $A.length$ 
5    $C[A[j]] = C[A[j]] + 1$ 
6 //  $C[i]$  now contains the number of elements equal to  $i$ .
7 for  $i = 1$  to  $k$ 
8    $C[i] = C[i] + C[i - 1]$ 
9 //  $C[i]$  now contains the number of elements less than or equal to  $i$ .
10 for  $j = A.length$  downto 1
11    $B[C[A[j]]] = A[j]$ 
12    $C[A[j]] = C[A[j]] - 1$ 
```

$\Theta(n)$ \nwarrow
 $K \leq n$

$\Theta(n+k)$
every Case

$A[j] \rightsquigarrow B[\text{?}]$

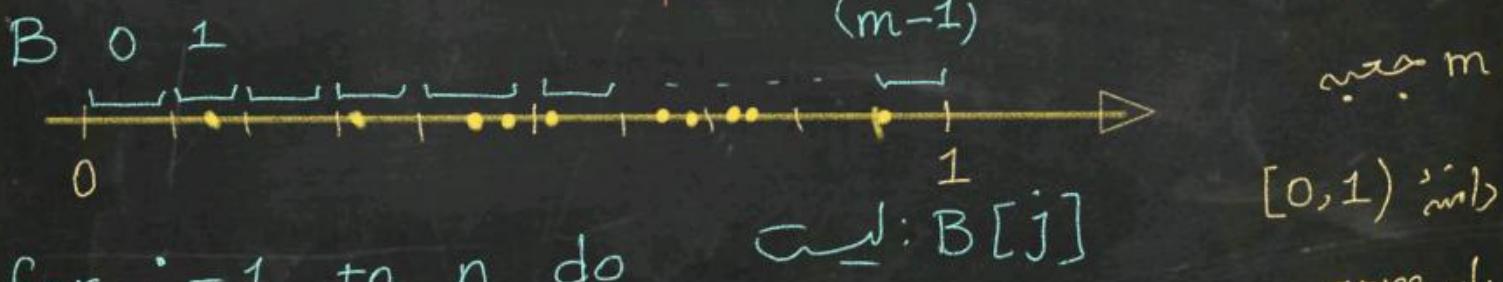
Bucket Sort

الگوریتم مرتب سازی جعبه ای

فرض کنید مجموعه ای از n عدد اعشاری در بازه $(0, 1)$ دارد.

A	1	$n = 10$
0.72 0.45 0.32 0.69 0.51 0.74 0.9 0.48 0.16 0.64		

می خواهیم این مجموعه را به ترتیب صعودی مرتب کنیم.



for $i = 1$ to n do

$\leftarrow B[j]$

$B[m * A[i]] \cdot \underline{\text{append}}(A[i])$

جعبه m

دسته $[0, 1)$

را به m جعبه افزایش کنیم.

($m = 10$)

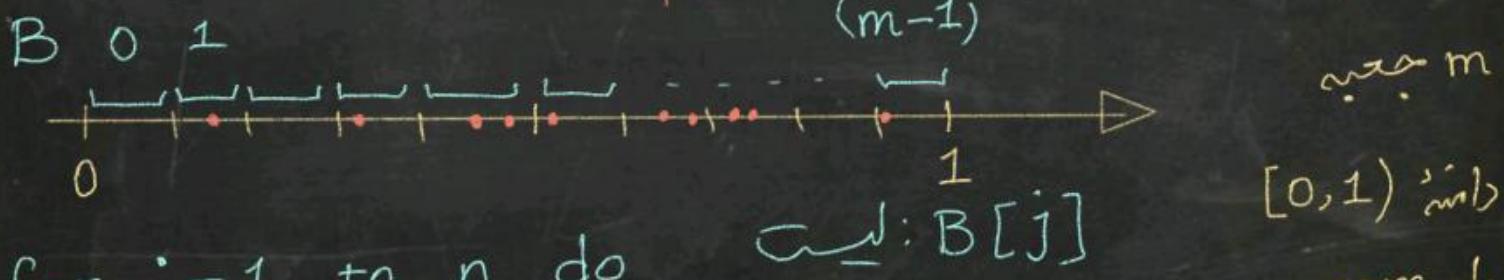
Bucket Sort

الگوریتم مرتب سازی جعبه ای

فرض کنید مجموعه ای از n عدد اعشاری در بازه $(0, 1)$ دارد.

A	1	$n=10$
0.72 0.45 0.32 0.69 0.51 0.74 0.9 0.48 0.16 0.64		

می خواهیم این مجموعه را به ترتیب صعودی مرتب کنیم.



for $i = 1$ to n do $\leftarrow B[j]$

$$B \left[\left\lfloor m * \frac{A[i] - \min}{\max - \min + \epsilon} \right\rfloor \right] \cdot \underline{\text{append}(A[i])}$$

را به m جعبه های اندازه افزایش کنیم.
($m=10$)

چنانچه بتوانیم این ورگشی را مفروض بگذاریم که برآورده داده شده به قسم اسکن
تعداد نقاط واقع شده در هر جعبه بطور میانگین α عدد نسبت متناظر باز n است

$$\alpha := \frac{n}{m} \quad \alpha \text{ is constant}$$

$$m = \frac{n}{\alpha} \quad (\text{متصل از } n)$$

لینی در هر جعبه بعد از میانگین تعداد نقطه (α) نظر داریم

for $j=0$ to $(m-1)$ do \rightarrow constant

insert-sort($B[j]$) $\rightsquigarrow O(\lceil \alpha \rceil^2)$

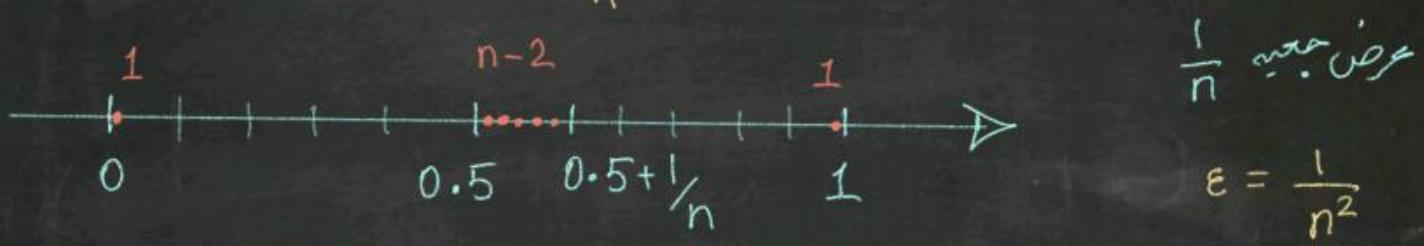
return concatenation($B[0], \dots, B[m-1]$)

هزینه زمان الگوریتم مرتب سازی جعبه ای (مرسانی) $O(n+m)$
 $n = \alpha m$ $O(n)$ \leftarrow $O(n+m)$

- در این حالت برای مرتب سازی جعبه‌ای \hat{A} ممکن است بجز نصف اول و آنرا در
 جعبه فارمی کنیم. مساله:

$$A = \left\langle 0, 0.5, 0.5 + \epsilon, 0.5 + 2\epsilon, \dots, 0.5 + \frac{1}{n} - \epsilon, 1 - \epsilon \right\rangle$$

$m = n$



در این حالت حداکثر زمان الگوریتم $O(n^2)$

- در این حالت، در هر جعبه حداقل n^2 نصیب و کار دارد.

- توجه: اگر معادل در بازه $[0, 1]$ می‌باشد باشد $\tilde{x} = \frac{x - \min}{\max - \min + \epsilon}$ بجزه $[0, 1]$.

الگوریتم مرتب سازی مبنایی

فرض نمایند معاویری که می خواهیم مرتب کنیم در یک مبنای عددologی ناچار داره سه داند.

مثال ۱. آرایه A را در صورتی که از مبنای عددologی که در مساله ۹ مذکور شده باشد در طرزی که در مساله ۹ مذکور شده باشد مرتب کنید.									
A	<table border="1"> <tr><td>441</td></tr> <tr><td>743</td></tr> <tr><td>905</td></tr> <tr><td>247</td></tr> <tr><td>615</td></tr> <tr><td>402</td></tr> <tr><td>213</td></tr> <tr><td>781</td></tr> </table>	441	743	905	247	615	402	213	781
441									
743									
905									
247									
615									
402									
213									
781									
	<table border="1"> <tr><td>441</td></tr> <tr><td>781</td></tr> <tr><td>402</td></tr> <tr><td>743</td></tr> <tr><td>213</td></tr> <tr><td>905</td></tr> <tr><td>615</td></tr> <tr><td>247</td></tr> </table>	441	781	402	743	213	905	615	247
441									
781									
402									
743									
213									
905									
615									
247									
	<table border="1"> <tr><td>441</td></tr> <tr><td>743</td></tr> <tr><td>213</td></tr> <tr><td>905</td></tr> <tr><td>615</td></tr> <tr><td>781</td></tr> <tr><td>247</td></tr> </table>	441	743	213	905	615	781	247	
441									
743									
213									
905									
615									
781									
247									
	<table border="1"> <tr><td>213</td></tr> <tr><td>247</td></tr> <tr><td>402</td></tr> <tr><td>213</td></tr> <tr><td>615</td></tr> <tr><td>441</td></tr> <tr><td>743</td></tr> <tr><td>781</td></tr> </table>	213	247	402	213	615	441	743	781
213									
247									
402									
213									
615									
441									
743									
781									

stable

sort
on
1st digit

stable

sort
on
2nd digit

stable

sort
on
3rd digit

بعد و فرض کنیم

طریق ناچار می باشد

عنصر A حفظ شود

برآید

(d=3 : مساله ۹)

- در الگوریتم مرتب‌سازی بینایی، برای $i=1, 2, \dots, d$ نزدیکه ارزش مکانی حرفم در ناچار اعداد است، هر بار عناصر A را براساس رقم i از میانه انتخاب کنیم.

- در اینجا از الگوریتم مرتب‌سازی سهارسانی به عنوان مرتب‌سازی مبنای اسعاد نامیدیم.

$O(n \cdot d + b \cdot d)$:
از اینجا هر سه زمان اجرای الگوریتم Radix Sort برابرست با:
for $i=1$ to d (فرض کنید n عدد و b مبنای d است)
using CountingSort sort A based on digit i
 $\rightarrow O(n+b) \times (d \text{ متراد})$
 $\rightarrow \text{const}$

Radix Sort

غیر مترقبه ای

$$O(n \cdot d) \quad \text{stable} \checkmark$$

HeapSort

بستگی داشته باشد

~~stable~~

$$O(n \log n)$$

طول n^b از اعداد در میانی b

$\{0, \dots, n\}$ مجموعه ای از اعداد صریح n

$$d = \lfloor \log_b n \rfloor$$

- توجه نماین که الگوریتم Radix Sort را برای مرتب سازی رشته های کاراکتری بحسب ترتیب لغات های نزدیک و آن برابر است.

- حالت اول (س). n رشته کاراکتری با حداکثر 4 مترشته ها

Ascii code	b = 256
"abcd"	↓
"xyzt"	↓
:	↓
:	↓
:	↓

لجهور میگان باید d است. $O(\sum_{i=1}^n \text{len}(s_i))$ *(نمیخواهد)*

- حالت دوم . n رشته کاراکتری با طولهای متساوی

"abx" > "acd" *'padding'*
"abx□" "acd" *with blank*