

11/18/2020

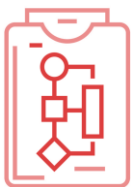


Homework 3

Sort & Ch9



ALGORITHM DESIGN



۱) نشان دهید زمان اجرای الگوریتم مرتب سازی سریع هنگامی که آرایه ورودی شامل اجزای متمایزی باشد که بصورت نزولی مرتب شده‌اند، برابر با $\Theta(n^2)$ خواهد بود.

میدانیم که کد quick sort بصورت زیر است:

```
quickSort(arr[], p, r)
    if (p < r)
        q = partition(arr, p, r)
        quickSort(arr, p, q - 1)
        quickSort(arr, q + 1, r)
```

و برای تابع partition داریم:

```
partition (arr[], p, r)
    pivot = arr[r]
    i = (p - 1)
    for (j = p; j <= r - 1; j++)
        if (arr[j] < pivot)
            i++
            swap arr[i] and arr[j]
    swap arr[i + 1] and arr[r]
    return (i + 1)
```

اگر آرایه از همان ابتدا sort شده باشد (چه بصورت صعودی و چه نزولی) در قسمت partition، عنصر pivot همواره در سر یا ته قرار خواهد گرفت. یعنی اگر آرایه ای n عنصری در اختیار داشته باشیم بعد از فراخوانی partition، در یک سمت pivot صفر عنصر و در سمت دیگر n-1 عنصر قرار میگیرد. یعنی:

$$T(n) = T(0) + T(n-1) + \Theta(n) = T(n-1) + \Theta(n)$$

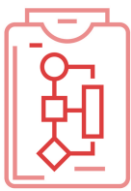
پس در هر مرحله باید دوباره روی n-1 عنصر باقی مانده تابع partition که خود $\Theta(n)$ میباشد را فراخوانی کنیم. بنابراین میتوان به این نتیجه رسید که در چنین حالتی الگوریتم $\Theta(n^2)$ خواهد بود.

۲) توضیح دهید که چگونه می توان n عدد صحیح در بازه ی 0 تا $n^2 - 1$ را در زمان $O(n)$ مرتب کرد.

میدانیم در الگوریتم radix sort، $O(d(k+n))$ زمان مصرف میشود. در این حالت k برابر است با تعداد حالاتی که یک رقم میتواند داشته باشد. همچنین d نیز تعداد ارقام اعدادی است که میخواهیم sort کنیم. با توجه به اینکه میخواهیم اعداد 0 تا $n^2 - 1$ را بررسی کنیم، میتوان با این فرض که اعداد را در مبنای n میبینیم گفت که قرار است اعدادی که مرتب میکنیم اعداد دو رقمی در مبنای n هستند. (مثلا برای $n = 3$ اگر اعداد 0 تا 9 را بنویسیم و آنها را به مبنای 3 تبدیل کنیم به ترتیب برابر خواهند بود با 00، 01، 02، 10، 11، 12، 20، 21 و 22)

یعنی قرار است n عدد $d = 2$ رقمی که ارقام آن میتواند $k = n$ هستند را با radix sort مرتب کنیم. یعنی:

$$O(d(k+n)) = O(2(n+n)) = O(2(2n)) = O(4n) = O(n)$$



۳) در الگوریتم SELECT، ورودی ها در ابتدا به گروه های ۵ تایی تقسیم می شوند. آیا در صورتی که ورودی ها به دسته های ۷ تایی تقسیم شوند الگوریتم همچنان در زمان خطی کار می کند؟

با تقسیم ورودی ها به ۷ قسمت، $\lceil \frac{n}{7} \rceil$ گروه خواهیم داشت. حداقل نیمی از این گروه ها، حداقل ۴ عضو بزرگتر از میانه ی میانه های این ۷ قسمت دارند و همچنین میانه ی میانه ها حداقل از میانه های ۴ گروه از $\lceil \frac{n}{7} \rceil$ کوچکتر می باشد. میتوان به نوعی گفت که تقریباً میانه ها از $\frac{4n}{14}$ عناصر ورودی بزرگتر است و از $\frac{4n}{14}$ عناصر ورودی کوچکتر می باشد. پس نمیتوان بیشتر از $\frac{10n}{14}$ بار آن را بصورت بازگشتی فراخوانی کرد. بنابراین خواهیم داشت $T(n) \leq T(\frac{n}{7}) + T(\frac{10n}{14}) + O(n)$. حال باید نشان دهیم این رابطه در زمان خطی کار میکند. با فرض $T(n) \leq cn$ برای $n < k$ خواهیم داشت:

$$\begin{aligned} T(n) &\leq T\left(\frac{n}{7}\right) + T\left(\frac{10n}{14}\right) + O(n) \\ &\leq c\left(\frac{n}{7}\right) + c\left(\frac{10n}{14}\right) + O(n) \\ &= cn\left(\frac{1}{7} + \frac{10}{14}\right) + O(n) = O(n) \end{aligned}$$

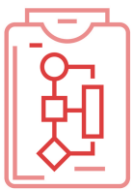
بنابراین این الگوریتم برای گروه های ۷ تایی نیز بصورت خطی کار میکند.

۴) فرض کنید می خواهیم n عدد صحیح را که شامل $\log n$ عدد متمایز است مرتب کنیم. الگوریتمی با پیچیدگی زمانی $O(n \log \log n)$ برای مرتب کردن مقایسه ای این اعداد طراحی کنید.

۵) فرض کنید که $L = (x_1, x_2, \dots, x_n)$ فهرستی از n عدد حقیقی باشد. ترتیب اعداد مشخص نیست. میگوییم عدد x در فهرست L «غالب» است اگر اکیداً بیشتر از $1/3$ اعداد در فهرست L مساوی x باشند. این (لینک) الگوریتم قطعی مبتنی بر مقایسه زیر را برای پیدا کردن اعداد «غالب» در نظر بگیرید:
الف) درستی الگوریتم و کد آن را ثابت کنید. (می توانید از loop invariant ها کمک بگیرید)
ب) ثابت کنید که این الگوریتم از $\theta(n)$ مقایسه انجام می دهد.

تکه کد این الگوریتم به شرح زیر است:

```
function(arr, n)
    count1 = 0
    count2 = 0
    first = sys.maxsize
    second = sys.maxsize
    for (i = 0; i < n; i++)
        if (first == arr[i]) count1 += 1
        else if (second == arr[i]) count2 += 1
        else if (count1 == 0)
            count1 += 1
            first = arr[i]
        else if (count2 == 0):
            count2 += 1
            second = arr[i]
        else
            count1 -= 1
            count2 -= 1
```



```

count1 = 0
count2 = 0
for (i = 0; i < n; i++)
    if (arr[i] == first) count1 += 1
    else if (arr[i] == second) count2 += 1
if (count1 > n / 3) return first
if (count2 > n / 3) return second
return -1

```

- الف)** می‌خواهیم اثبات کنیم که for اول، دو انتخاب مربوط به عنصر غالب را در صورت وجود آن درست انجام می‌دهد.
- **Loop Invariant:** در پیمایش آرایه هنگامی که در حال بررسی عنصر i هستیم دو عنصر نامزدی که بیشترین تکرارها را در بازه $arr[0]$ تا $arr[i]$ دارند انتخاب شده‌اند.
 - **Initialization:** در ابتدا i برابر با ۰ است. در تکرارهای ۰ و ۱ با توجه با مقادیر count در دو متغیر first و second برای اولین بار مقدار دهی میشوند.
 - **Maintenance:** در هر مرحله دو کاندید و مقادیر تکرار هر کدام تا اینجا مشخص شده‌اند. با رفتن به پیمایش بعدی آرایه، یا مقادیر تکرار هر کاندید عوض میشوند یا حالتی وجود دارد که کاندید قبلی مقدار ۰ دارد و خود کاندید آپدیت میشود.
 - **Termination:** در آخرین مرحله تکرار که i با $n-1$ برابر است، دو نامزدی که در احتمال دارد عناصر غالب باشند مشخص شده‌اند و در ادامه میتوان چک کرد که آیا واقعا قالب هستند یا خیر.

ب) در کدی که در بالا آمده داریم:

```

function(arr, n)
:
for (i = 0; i < n; i++)
:
:
for (i = 0; i < n; i++)
:
:

```

در این کد که خلاصه شده ی کد قبل تر میباشد، میتوان گفت در کل ۲ حلقه for داریم که کل عناصر آرایه ی مورد نظر را طی میکنند و مقایسه هایی انجام میدهند. قسمت هایی که با علامت : به اختصار در آمده‌اند، اعمالی نظیر چک کردن شروط مختلف (if...else...)، مقدار دهی ها و همچنین return میباشند که هر کدام از آنها دارای $\Theta(1)$ هستند. بنابراین محتوای داخل حلقه های for نیز $\Theta(1)$ مقایسه دارند. پس میتوان گفت هر یک از حلقه ها دارای $\Theta(n)$ مقایسه هستند. از طرفی میدانیم مقایسه های بعد از for دوم نیز همگی $\Theta(1)$ هستند. با توجه به تمام این موضوعات میتوان ادعا کرد:

$$\Theta(1) + 2\Theta(n) = \Theta(1) + \Theta(n) = \Theta(n)$$

- مهلت ارسال تمرین ساعت ۲۳:۵۵ روز **جمعه ۷ آذرماه** می‌باشد.
- سوالات خود را می‌توانید از طریق ایمیل از تدریس‌یاران بپرسید.
 - parsa.abdollahi.pa@gmail.com
 - faridi.mina.1@gmail.com
- ارائه پاسخ تمرین به دو روش ممکن است:
 - (۱) تایپ داخل همین فایل و ارائه فایل Pdf
 - (۲) چاپ تمرین و پاسخ دهی به صورت دستنویس خوانا
- ارائه تمرین به روش اول شامل ۱۰٪ نمره امتیازی می‌گردد.
- فایل پاسخ تمرین را تنها با قالب **HW3-9531747.pdf** در مدل بارگزاری کنید.
- فایل زیپ ارسال نکنید.