دانشکده علوم ریاضی و آمار

مدرس: دکتر مجتبی رفیعی

نیمسال اول ۱۴۰۰-۱۴۰۱

ساختمان داده‌ها و الگوریتم‌ها

جلسه ۱۳ ساختمان داده‌ها و الگوریتم‌ها

نگارنده: فاطمه خورسند

۹ آبان ۱۴۰۰

فهرست مطالب

۱ مرور مفاهیم جانبی

فرض کنید $f(n) = logn!$ و $g(n) = n\log n$ باشد. نشان دهید که $f(n) = \theta\big(g(n)\big)$ می‌باشد.
طبق تعریف نماد مجانبی θ، می‌دانیم که

$$f(n) = \theta\big(g(n)\big) \iff f(n) = O(g(n)) \quad \text{and} \quad f(n) = \Omega(g(n))$$

نماد O را قبلا نشان داده‌ایم. طبق تعریفی که برای Ω گفتیم،

$$\exists\, c, n_0 \quad s.t. \quad \forall n \geq n_0 \quad , \quad c \times n\log n \leq \log n! \quad (*)$$

راهنمایی:

$$n! = \sqrt{2\pi n}\left(\frac{n}{e}\right)^n e^{\alpha_n}$$

$$\frac{1}{12n+1} \leq \alpha_n \leq \frac{1}{12n}$$

از دو طرف لگاریتم می‌گیریم و داریم:

$$\log n! = \frac{1}{2}\log(2\pi n) + n\log(\frac{n}{e}) + \alpha_n \log e$$

$$= \frac{1}{2}\log(2\pi) + \frac{1}{2}\log n + n\log n - n\log e + \alpha_n \log e \ \geq c \times n\log n$$

$\alpha_n \log e$ و $\frac{1}{2}\log(2\pi)$ مثبت و ثابت $(constant)$ هستند. پس داریم:

$$A = c + \frac{1}{2}\log n + n\log n - n\log e = O(n\log n)$$

یعنی:

$$\exists\ n_0', d \quad s.t. \quad \forall n \geq n_0' \qquad A \leq d\, n\log n$$

$$(*) \quad \rightarrow \quad c \times n\log n \leq d\, n\log n$$
$$d = c+1$$
$$n_0' = n_0$$

۲ مرتب‌سازی حبابی

انواع مرتب‌سازی‌هایی که تاکنون مورد بررسی قرار دادیم، عبارتند از:

۱. مرتب‌سازی درجی $(Insertion\ sort)$،

۲. مرتب‌سازی سریع $(Quick\ sort)$،

۳. مرتب‌سازی حبابی $(Bubble\ sort)$،

مرتب‌سازی حبابی: با ذکر مثال این مرتب‌سازی را شرح می‌دهیم. آرایه $A = 4, 2, 7, 5, 6, 1$ را در نظر بگیرید. در هر مرحله سعی می‌شود، دو عنصر آخر مرتب شود و عنصر کوچکتر در سمت چپ قرار گیرد و به همین ترتیب عنصر اولیه از ابتدا، دوتا دوتا با بقیه عناصر مقایسه می‌شود. در مرحله اول عنصر مینیمم در آرایه پیدا و در اولین خانه قرار گرفت.

مرحله ۱:

- $4, 2, 7, 5, \underline{6, 1}$
- $4, 2, 7, \underline{5, 1}, 6$
- $4, 2, \underline{7, 1}, 5, 6$
- $4, \underline{2, 1}, 7, 5, 6$
- $\underline{4, 1}, 2, 7, 5, 6$
- $\underline{1}, 4, 2, 7, 5, 6$

مرحله ۲:

- $1, 4, 2, 7, \underline{5, 6}$

- $1, 4, 2, \underline{7, 5}, 6$
- $1, 4, \underline{2, 5}, 7, 6$
- $1, \underline{4, 2}, 5, 7, 6$
- $1, \underline{2}, 4, 5, 7, 6$

در آخر مرحله دوم دو عنصر مرتب داریم. همین ترتیب را تا آخرین مرحله انجام می‌دهیم.

مرحله ۵:

- $1, 2, 4, 5, \underline{6, 7}$
- $1, 2, 4, 5, \underline{6}, 7$

قبل از شروع مرحله ۵، تعداد ۴ عنصر مرتب شده است. از آخر آرایه دوتا دوتا مقایسه می‌شوند و مقدار مینیمم شیفت می‌خورد تا در خانه اول (قسمت مرتب نشده) قرار گیرد. در مرحله پنجم تمام آرایه‌ها مرتب شده‌اند. درواقع برای مرتب شدن آرایه‌ها به $(n-1)$ مرحله نیاز است.
نکته: در پایان مرحله i _ام تعداد i عنصر ابتدایی مرتب می‌شوند.

۳ الگوریتم مرتب‌سازی حبابی

شبه کد مربوط به الگوریتم مرتب‌سازی درجی در ادامه آورده شده است.

Algorithm 1 Bubble-sort(A[1..n])

```
1: for i = 1 to n − 1 do
2:     for j = n downto i + 1 do
3:         if (A[j] < A[j − 1]) then
4:             swap(A[j], A[j − 1])
```

این الگوریتم، یک الگوریتم درجاست چرا که حافظه اضافی که برحسب اندازه ورودی باشد نیاز ندارد و روی خود آرایه A کار می‌کند.

یادآوری- اثبات درستی

۱: تعیین خصیصه ناوردایی،

۲: طی کردن گام‌ها:

(آ) گام آغازین،
(ب) گام نگهداری،
(ج) گام پایان.

$LI1$ و $LI2$ را داریم، که $LI1$ حلقه بیرونی for (خطوط ۱ تا ۴) است و $LI2$ حلقه درونی for (خطوط ۲ تا ۴) است.

حلقه درونی $LI2$: باید خصیصه ناوردایی را درست انتخاب کنیم. در مرحله j-ام عنصر مینیمم در زیرآرایه $A = [j..n]$ در خانه j-ام قرار می‌گیرد. در شروع حلقه $j = n$ است، یعنی اگر $A[n..n]$ را در نظر بگیریم چون یک خانه است، پس شرط اجرا نمی‌شود. چون عنصر مینیمم در خود آن است و مقایسه‌ای صورت نمی‌گیرد. در ادامه، j را یکی کم می‌کنیم و $n-1$ می‌شود. حال باید n را با $n-1$ مقایسه کنیم و عنصر مینیمم در خانه $n-1$ قرار گیرد. به همین ترتیب پیش می‌رویم تا به شرط پایان حلقه که $j = i$ است برسیم. نتیجه می‌گیریم از 1 تا n در آرایه، کوچکترین عنصر در خانه i قرار گرفته است.

حلقه بیرونی $LI1$: شبیه قبل، خصیصه ناوردایی و گام‌های ناوردایی را می‌بایست تعیین کنیم:
خصیصه ناوردایی: در پایان مرحله i _ ام، i عنصر ابتدایی آرایه A یعنی $A[1..i]$ مرتب می‌شوند.

- **گام آغاز:** $i = 1$ می‌باشد، پس $A[1..1]$ یک خانه دارد، پس مرتب است.
- **گام نگهداری:** در مرحله $i = k$ می‌دانیم از 1 تا $k - 1$ مرتب است و در پایان آن مرحله $A[1..k]$ مرتب است.
- **گام پایانی:** $i = n$ می‌باشد، پس $A[1..n]$ مرتب است.