Subject: جلسه دوم
Year. ۱۴۰۰ Month. ۶ Date. ۳۰

موضوع: تحلیل بهترین حالت، بدترین حالت و حالت متوسط

مثال ۱) [جستجو]: آرایه A شامل جایگشتی از اعداد ۱ تا n داده شده است. همچنین عدد x نیز داده شده است. خروجی: اندیس i به گونه‌ای که A[i]=x است. $1 \le x \le n$

ورودی: n تا عدد و یک عدد x. هدف تعیین زمان اجرای الگوریتم برحسب n

راه حل ۱) جستجوی خطی یا sequential search

```
for (i=1 → n)
    if (A[i] = x)
        return i;
```

سوال: تعداد مقایسه‌های الگوریتم (خط +) چند تاست؟

هیچکس الگوریتم در بهترین حالت بررسی نمی‌کند. Best برای مشخص بودن حداقل است. $n(1) = 1$ — Best ۱ مقایسه

در computer science معمولاً بدترین حالت را بررسی می‌کند. $O(1) = n$ — worst n مقایسه

Average

* دقت کنید که برای محاسبه حالت متوسط، نیاز به داشتن در رابطه با توزیع ورودی هستیم

مثلاً فرض کنید بدانیم که x با احتمال یکسان می‌تواند هرکدام از اعضا باشد. با احتمال $\frac{1}{n}$ عنصر اول آرایه

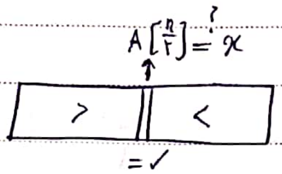
$$\left.\begin{array}{l} \frac{1}{n} \to 1 \\ \frac{1}{n} \to 2 \\ \frac{1}{n} \to 3 \\ \vdots \\ \frac{1}{n} \to n \end{array}\right\} \quad \frac{1}{n}[1+2+3+\dots+n] = \frac{1}{n}\cdot\frac{n(n+1)}{2} = \frac{n+1}{2} \simeq O(n)$$

Subject: جلسه دوم
Year. ۱۴۰۰ Month. ۶ Date. ۳۰ ( )

مسئله ۲ : [جست‌وجو در آرایه‌ی مرتب شده]   ورودی: آرایه‌ی A شامل n عدد و عنصر x $x \in A$
مرتب شده   فرض $n = 2^k - 1$

خروجی: اندیس i به صورتی که $A[i] = x$

۱. روش جست‌وجوی خطی
۲. روش جست‌وجوی دودویی

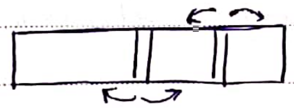


bin_search(1, n)

```
binary_search (low, high)
  mid = (low + high)/2
  if (A[mid] > x)
      return (bin_search(low, mid-1))
  if (A[mid] < x)
      return (bin_search(mid+1, high))
  else
      return mid;
```

تحلیل

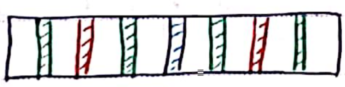
Best case : ۱ مقایسه = $\Omega(1)$



worst case : $n = 2^k - 1$ $k = \log_2^{(n+1)}$

مبنا لگاریتم ۲ است اما در کل مبنا مهم نیست ←

Average case : فرض: عدد x با احتمال یکسان در هر کدام از اعضای آرایه می‌تواند باشد



$$\frac{1}{n}\times 1 + \frac{2}{n}\times 2 + \frac{4}{n}\times 3 + \frac{8}{n}\times 4 + \dots + \frac{\frac{n+1}{2}}{n}\log^{(n+1)}$$

$$\frac{1}{n}\left[1 + 2\times 2 + 4\times 3 + 8\times 4 + \dots + \frac{n+1}{2}\times \log(n+1)\right] = T$$

Subject: جلسه دوم
Year. ۱۴۰۰ Month. ۶ Date. ۳۰ ( )

برای مقدار T حد پایین و بالا را بدست می آوریم.

نصف عددها را در نظر می گیریم

حد پایین: $\frac{1}{n} \times \left[\frac{n}{2} \times \log(n+1)\right] \leq T$

$\frac{\log(n+1)}{2} \leq T$

حد بالا $T \leq \frac{1}{n}\left[1 \times \log(n+1) + 2 \times \log(n+1) + 4 \times \log(n+1) + \dots + \frac{n+1}{2}(\log(n+1))\right]$

$T \leq \frac{1}{n} \log(n+1) \left[\underbrace{1+2+4+\dots+\frac{n+1}{2}}_{n}\right]$

$T \leq \log(n+1)$

$T = \Theta(\log n)$

مثال ۳ [مرتب سازی]: ورودی آرایه A شامل n عدد
خروجی آرایه A به صورت مرتب شده

* الگوریتم ۱: مرتب سازی حبابی (Bobble_Sort)

A: ۵ ۶ ۲ ۴ ۱
۵ ۲ ۶ ۴ ۱
۵ ۲ ۴ ۶ ۱
۵ ۲ ۴ ۱ ۶
۲ ۵ ۴ ۱ ۶
۲ ۴ ۵ ۱ ۶
۲ ۴ ۱ ۵ ۶

برای n عدد حداکثر n-1 مقایسه انجام می شود.

سوال: چرا الگوریتم درست است؟

در Pass اول، بزرگترین عنصر به انتهای آرایه می رود

در Pass دوم، دومین بزرگترین عنصر به جایگاه n-1 ام می رود.

⋮

در n-1 پس آرایه مرتب می شود.

Subject: جلسه دوم
Year. ۱۴۰۰ Month. ۶ Date. ۳۰ ( )

تعداد مقایسه ها در بهترین، بدترین و حالت متوسط حساب کنید. (فرض: در هر pass اگر swap نداشتیم ← متوقف می شویم)

بهترین حالت: زمانی که آرایه مرتب باشد: ۱ تا n-1 مقایسه: $\Omega(n)$

بدترین: زمانی که آرایه نزولی باشد: $n(n-1) = n^2 - n = O(n^2)$

حالت متوسط: هدف: پیدا کردن جوابی برای متوسط تعداد مقایسه ها. تعداد swap ≠ تعداد مقایسه

* متوسط تعداد swap را پیدا کنیم؟

تعریف) نابه‌جایی یا inversion: جفت اندیس از i و j به گونه‌ای که $i < j$ و $A[i] > A[j]$

* تعداد نابجایی ها در هر swap ۱ واحد کم می شود.
* آخر کار تعداد نابجایی ها = ۰

بنابراین ← تعداد swap = تعداد نابجایی ها

← متوسط تعداد نابجایی ها؟ فرض: همه n! جایگشت مختلف احتمال یکسان دارند.

$$\frac{\frac{n!}{2} \times \binom{n}{2}}{n!} = \frac{\binom{n}{2}}{2} = \frac{n(n-1)}{4} = \Omega(n^2)$$