Medians and Order Statistics

- ith order statistic ، i امین عنصر کوچکتر در یک مجموعه n عنصری از اعداد است .
- minimum ، در واقع first order statistic است. $i = 1$
- maximum ، در واقع nth order statistic است. $i = n$
- median ، در واقع halfway point در یک مجموعه است.
- وقتی n فرد باشد، median یک است و در $i = \frac{n+1}{2}$ قرار دارد.
- وقتی n زوج باشد، دو تا median وجود خواهد داشت :
  - The lower median, at $i = n/2$
  - The upper median, at $i = n/2 + 1$
- منظور از median را در این حالت lower median در نظر می‌گیریم.

The selection problem

Input : A set A of n distinct numbers and a number i with $1 \leq i \leq n$.

Output : The element $x \in A$ that is larger than exactly $i-1$ other elements in A, in other word, the ith smallest element of A.

مسئله selection می‌تواند در زمان $O(n\lg n)$ حل شود

- اعداد را با استفاده از یک الگوریتم با زمان $O(n\lg n)$ مرتب می‌کنیم مانند heapsort یا merge sort.
- عنصر ith را در آرایه sort شده، برمی‌گردانیم.

الگوریتمهای سریعتر وجود دارد.

- ابتدا، به بررسی الگوریتمهای یافتن maximum و minimum می‌پردازیم که بر روی یک مجموعه از n عنصر کار می‌کنند.
- سپس نگاهی به یک الگوریتم ساده کلی selection می‌اندازیم با یک محدودیت زمانی $O(n)$ در average case.
- در پایان، نگاهی به یک الگوریتم بیشتر پیچیده و کلی برای selection داریم که در حالت worst case در $O(n)$ کار می‌کند.

Minimum and maximum:

به راحتی یک upper bound برای تعداد comparisons ها برای این مسئله برابر با $n-1$ بدست می‌آید وقتی به دنبال یافتن minimum در یک مجموعه n تایی هستیم.

- تمام عناصر را بررسی کن به نوبت و کوچکترین آنها را نگه دار.
- این بهترین کاری است که می‌توان انجام داد. زیرا تمام عناصر به جز minimum باید با عناصر کوچکتر حداقل یک مرتبه مقایسه شوند.

در ادامه pseudo code زیر عنصر minimum را در آرایه A [1..n] می یابد.

```
MINIMUM (A, n)
min ← A[1]
for i ← 2 to n
    do if min > A[i]
        then min ← A[i]
return min
```

عنصر maximum هم می تواند به شیوه‌ی مشابهی بدست آید تنها کافی است در بالا به جای > علامت < استفاده شود.

Simultaneous minimum and maximum

(همزمانی یافتن کمترین و بیشترین)

در بسیاری از کاربردها نیازمند این هستیم که هر دوی minimum و maximum را همزمان در یک مجموعه عناصر بدست آوریم.

scale

-- به عنوان مثال یک graphic program ممکن است نیاز به این داشته باشد که یک مقیاس از یک مجموعه در (x, y) را در یک مستطیل نشان دهد.

- برای این منظور برنامه نیاز دارد که ابتدا minimum و maximum هر coordinate را مشخص کند.

یک الگوریتم ساده برای اینکار می تواند این باشد که min و max هر کدام مستقلاً محاسبه شوند در این حالت n-1 مقایسه برای min و n-1 مقایسه برای max وجود دارد.

در مجموع $2n-2$ مقایسه انجام می‌شود که زمان اجرای کل $\Theta(n)$ است.

در واقع، حداکثر $3\lceil n/2 \rceil$ مقایسه برای یافتن هر دوی آنها نیاز است (هم maximum وهم minimum)

- نگهداری هم minimum و هم maximum عناصر که تاکنون دیده شده‌اند.
- هر عنصر را مستقلاً با minimum و maximum مقایسه نکن.
- عناصر را جفتی مقایسه کن.
- عناصر هر جفت را با یکدیگر مقایسه کن.
- آن گاه عنصر بزرگتر را با maximum مقایسه کن که تاکنون بدست آمده و عنصر کوچکتر را با minimum که تاکنون بدست آمده مقایسه کن.

این باعث می‌شود ۳ تا مقایسه داشته باشیم برای هر ۲ عنصر از ورودی.

بسته به اینکه n فرد یا زوج باشد، مقداردهی اولیه به min و max را بررسی می‌کنیم.

- اگر n زوج باشد، دو عنصر اول را مقایسه کن و بزرگتر را max و کوچکتر را min در نظر بگیر و مابقی عملیات را جفتی ادامه بده.

- اگر n فرد باشد، هر دوی min و max را عنصر اول در نظر بگیر و مابقی عملیات را جفتی ادامه بده.

Analysis of the total number of comparisons

اگر n زوج باشد، یک مقایسه اولیه داریم و سپس $\frac{3(n-2)}{2}$ مقایسه دیگر.

$$\# \text{ of comparisons} = \frac{3(n-2)}{2} + 1$$

$$= \frac{3n-6}{2} + 1$$

$$= \frac{3n}{2} - 3 + 1$$

$$= \frac{3n}{2} - 2$$

اگر n فرد باشد، $\frac{3(n-1)}{2} = 3\lfloor \frac{n}{2} \rfloor$ مقایسه انجام می دهیم.

در هر دو حالت:

بیشترین تعداد مقایسه‌های انجام شده کوچکتر یا مساوی $3\lfloor \frac{n}{2} \rfloor$ است.

در ادامه در مورد Selection in expected linear time صحبت می کنیم