پیچیدگی الگوریتم مرتب سازی سریع

بررسی پیچیدگی الگوریتم مرتب سازی سریع (quick sort) :

- **بدترین حالت الگوریتم،** زمانی است که آرایه بعد از افراز، تقریبا به بخش هایی با اندازه 0 و $n-1$ تقسیم شود (به عبارتی، آرایه از قبل به صورت صعودی یا نزولی است):

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

طوری که $T(0)$, $T(n-1)$ و $n-1$ به ترتیب برابر زمان لازم برای مرتب سازی عناصر سمت چپ محور، زمان لازم برای مرتب سازی عناصر سمت راست محور و زمان مورد نیاز افراز میباشند.
بایستی فرمول بازگشتی حاصل را حل کنیم:

$$T(n) = T(n-1) + n-1$$

که این رابطه بازگشتی را به کمک روش تکرار و جایگذاری حل خواهیم کرد :

$$\begin{aligned} T(n) = T &= (n-1) + n - 1 \\ &= (T(n-2) + n - 2) + n - 1 = T(n-2) + (n-2) + (n-1) \\ &= (T(n-3) + n - 3) + (n-2) + (n-1) \\ &= T(n-3) + (n-3) + (n-2) + (n-1) \\ &= \text{After k steps} \\ &= T(n-k) + (n-k) + (n-k+1) + \ldots + (n-2) + (n-1) \end{aligned}$$

حالا اگر $n - k = 0$ باشد در این صورت، $n = k$ بوده و خواهیم داشت :

$$\begin{aligned} &= T(0) + (0 + 1 + 2 + \ldots + n - 1) \\ &= 0 + \frac{n(n-1)}{2} \in O(n^2) \end{aligned}$$

- **بهترین حالت الگوریتم**، زمانی است که آرایه مورد نظر پس از افراز، تقریبا به دو بخش مساوی تقسیم میشود. در این صورت زمان لازم برای اجرای الگوریتم با رابطه بازگشتی زیر قابل توصیف است که در آن $2T(n/2)$ زمان لازم برای مرتب سازی دوبخش آرایه و $n-1$ زمان مورد نیاز برای افراز آرایه است:

$$T(n) = \begin{cases} 0, n = 1 \\ 2T\left(\frac{n}{2}\right) + n - 1, n > 1 \end{cases}$$

فرمول بازگشتی $T(n)$ را با قضیه اصلی در طراحی الگوریتم حل میکنیم :

$a = 2$
$b = 2$
$f(n) = n - 1$
$f(n) = n - 1 = \Theta(n) = \Theta\left(\mathrm{n}^{\log_2 2}\right) = \Theta\left(\mathrm{n}^{\log_b a}\right)$
$\text{-->}\ \mathrm{T(n)} = \Theta\left(\mathrm{n}^{\log_b a}.\log n\right) = \Theta(\mathrm{n}.\log n) = \Omega(\mathrm{n}.\log n)$