ORDER STATISTICS - MEDIAN FINDING, RANK SELECTION

Given n unsorted elements, find the k-th smallest.

ORDER STATISTICS - MEDIAN FINDING, RANK SELECTION

Given n unsorted elements, find the k-th smallest.

↳ easy $O(n)$ if $k = O(1)$ or $n - O(1)$.
In some sense median is hardest

ORDER STATISTICS - MEDIAN FINDING, RANK SELECTION

Given n unsorted elements, find the k-th smallest.

↳ easy $O(n)$ if $k = O(1)$ or $n - O(1)$.

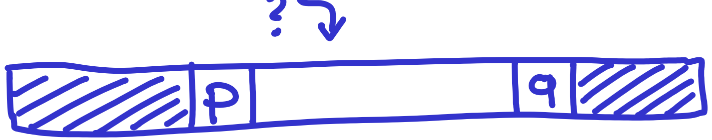
In some sense median is hardest

We will assume distinct elements.

RandSelect(p, q, i) // normally written ~(A, p, q, i)

↳ points to Array

?



↳ a randomized algorithm for finding the i-th smallest element between index p and index $q \geq p$ in an array

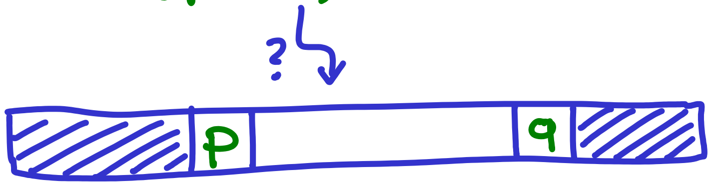
ORDER STATISTICS - MEDIAN FINDING, RANK SELECTION

Given n unsorted elements, find the k-th smallest.

↳ easy $O(n)$ if $k = O(1)$ or $n - O(1)$.
In some sense median is hardest

We will assume distinct elements.

RandSelect(p, q, i) // normally written $\sim (A, p, q, i)$
↳ points to Array



↳ a randomized algorithm for finding the i-th smallest element between index p and index $q \geq p$ in an array

Start out with RandSelect$(1, n, k)$

//Looking for i-th smallest element in [p...q]

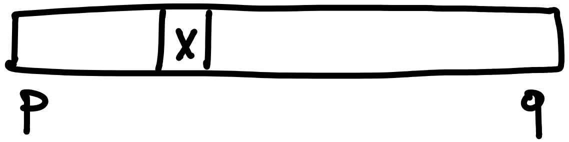
RandSelect(p, q, i)

// Looking for i-th smallest element in [p...q]

RandSelect(p,q,i)

$r \leftarrow$ RandPartition(p,q)

a random element



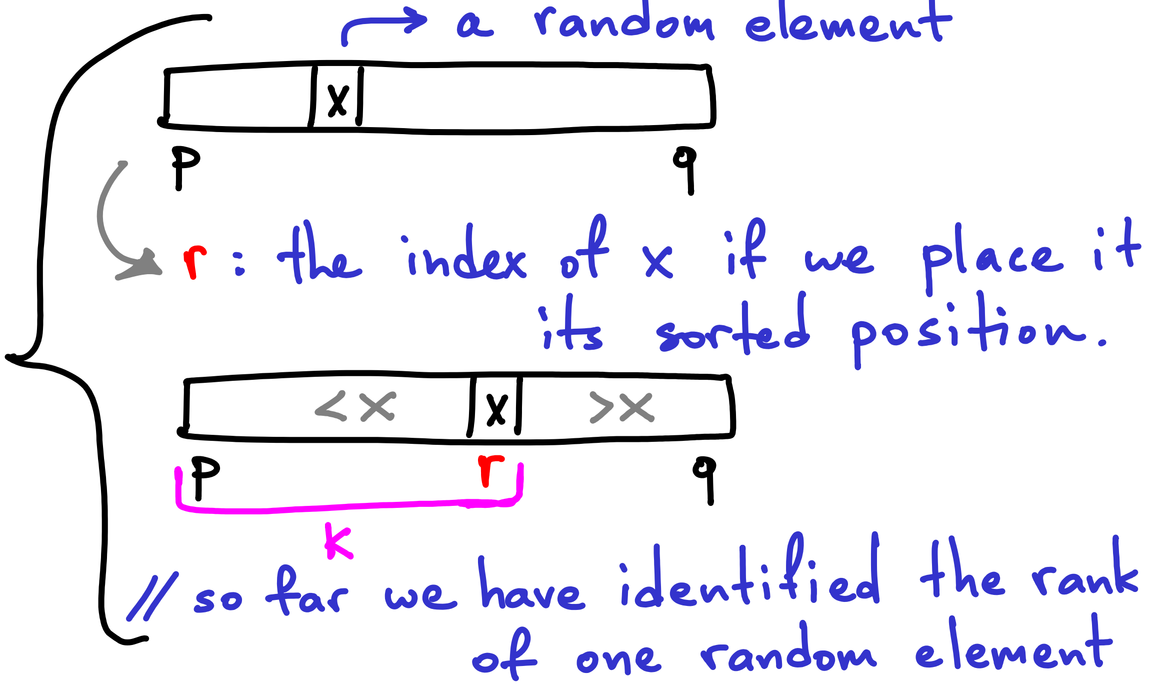
r : the index of x if we place it its sorted position.

// Looking for i-th smallest element in [p...q]

RandSelect(p, q, i)

  r ← RandPartition(p, q)

  k = r - p + 1 // rank of x in [p...q]



// Looking for i-th smallest element in [p...q]

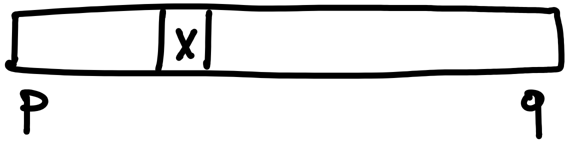
RandSelect(p,q,i)

$r \leftarrow$ RandPartition(p,q)

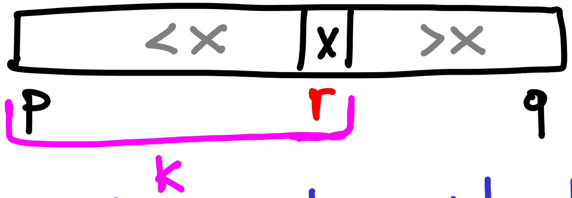
$k = r - p + 1$ // rank of x in [p...q]

if $i = k$, return A[r]

a random element



r : the index of x if we place it its sorted position.



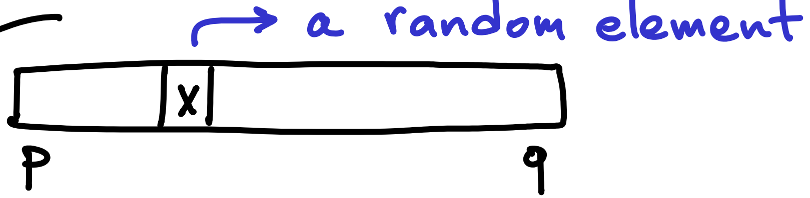
// so far we have identified the rank of one random element

// Looking for i-th smallest element in [p...q]

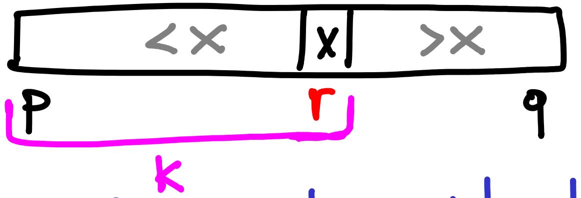
RandSelect(p, q, i)

$r \leftarrow$ RandPartition(p, q)

$k = r - p + 1$ // rank of x in [p...q]



r : the index of x if we place it its sorted position.



// so far we have identified the rank of one random element

if $i = k$, return A[r]

if $i < k$ ...

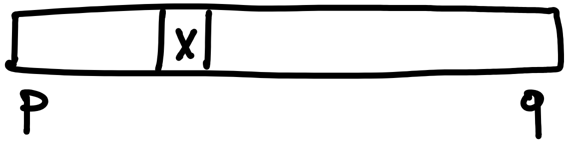
// Looking for i-th smallest element in [p...q]

RandSelect(p,q,i)

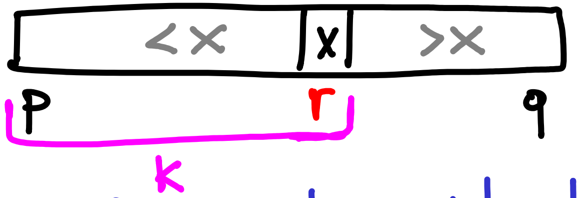
$r \leftarrow$ RandPartition(p,q)

$k = r - p + 1$ // rank of x in [p...q]

a random element



r : the index of x if we place it its sorted position.



// so far we have identified the rank of one random element

if $i=k$, return A[r]

if $i<k$, RandSelect(p, r-1, i) // we know that x has higher rank than i and all elements in left sub-array have smaller rank.

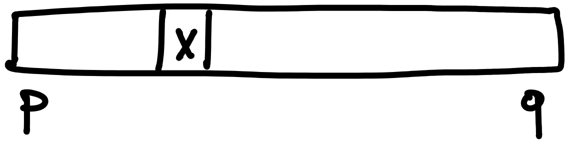
// Looking for i-th smallest element in [p...q]

RandSelect(p, q, i)

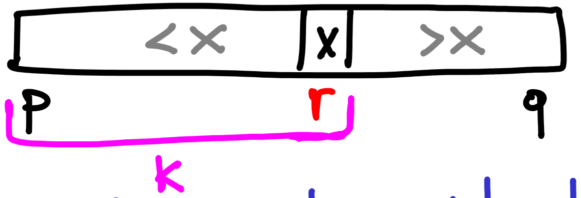
$r \leftarrow$ RandPartition(p, q)

$k = r - p + 1$ // rank of x in [p...q]

a random element



r : the index of x if we place it its sorted position.



// so far we have identified the rank of one random element

if $i = k$, return A[r]

if $i < k$, RandSelect(p, r-1, i) // we know that x has higher rank than i and all elements in left sub-array have smaller rank,

if $i > k$ ...

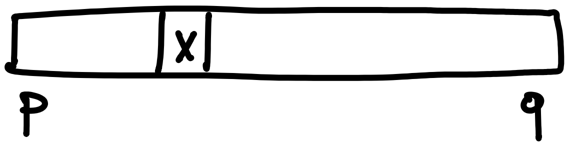
// Looking for i-th smallest element in [p...q]

RandSelect(p, q, i)

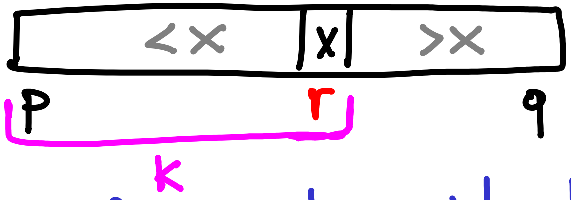
$r \leftarrow$ RandPartition(p, q)

$k = r - p + 1$ // rank of x in [p...q]

a random element



r : the index of x if we place it its sorted position.



// so far we have identified the rank of one random element

if $i = k$, return A[r]

if $i < k$, RandSelect(p, r-1, i) // we know that x has higher rank than i and all elements in left sub-array have smaller rank.

if $i > k$, RandSelect(r+1, q, i-k) // symmetric case. All that changes is that we "reset" the value of the target rank.

$$\begin{bmatrix} 6 & 10 & 13 & 5 & 8 & 3 & 2 & 11 \end{bmatrix} \qquad n=8 \qquad i=7$$

$[6 \quad 10 \quad 13 \quad 5 \quad 8 \quad 3 \quad 2 \quad 11]$ $n=8$ $i=7$

$r \leftarrow \text{RandPartition}(1, 8)$ $\rightarrow$ Let's just "randomly" pick $x = A[1] = 6$

$[6 \quad 10 \quad 13 \quad 5 \quad 8 \quad 3 \quad 2 \quad 11]$ $n=8$ $i=7$

$r \leftarrow \text{RandPartition}(1, 8)$ $\rightarrow$ Let's just "randomly" pick $x = A[1] = 6$

After $\Theta(n)$ work: $[5 \quad 3 \quad 2 \quad 6 \quad 10 \quad 13 \quad 8 \quad 11]$

$\llcorner r = 4 = k$

$[6 \quad 10 \quad 13 \quad 5 \quad 8 \quad 3 \quad 2 \quad 11]$ $n=8$ $i=7$

$r \leftarrow \text{RandPartition}(1,8)$ → Let's just "randomly" pick $x = A[1] = 6$

After $\Theta(n)$ work: $[5 \quad 3 \quad 2 \quad 6 \quad 10 \quad 13 \quad 8 \quad 11]$

$r = 4 = k$

We have found the 4-th smallest element, so the 7-th smallest is in $[10 \quad 13 \quad 8 \quad 11]$

$[6 \quad 10 \quad 13 \quad 5 \quad 8 \quad 3 \quad 2 \quad 11]$ $\quad n=8 \quad i=7$

$r \leftarrow$ RandPartition$(1, 8)$ $\rightarrow$ Let's just "randomly" pick $x = A[1] = 6$

After $\Theta(n)$ work: $[5 \quad 3 \quad 2 \quad 6 \quad 10 \quad 13 \quad 8 \quad 11]$

$r = 4 = k$

Recursively call RandSelect$(5, 8, 3)$ $\Leftarrow$ new $i = 7-4$

We have found the 4-th smallest element, so the 7-th smallest is in $[10 \quad 13 \quad 8 \quad 11]$

[6 10 13 5 8 3 2 11] $n=8$ $i=7$

$r \leftarrow$ RandPartition(1, 8) $\rightarrow$ Let's just "randomly" pick $x = A[1] = 6$

After $\Theta(n)$ work: [5 3 2 6 10 13 8 11]

$\lfloor r = 4$ $= k$

new $i = 7 - 4$

Recursively call RandSelect(5, 8, 3) $\Leftarrow$ We have found the 4-th smallest element, so the 7-th smallest is in [10 13 8 11]

Randpartition(5, 8) $\rightarrow$ Again pick $A[5] = 10$ // first parameter of function

[6 10 13 5 8 3 2 11] $n=8$ $i=7$

$r \leftarrow$ RandPartition(1, 8) $\rightarrow$ Let's just "randomly" pick $x = A[1] = 6$

After $\Theta(n)$ work: [5 3 2 6 10 13 8 11]

$r = 4 = k$

Recursively call RandSelect(5, 8, 3)   new $i = 7-4$ $\Leftarrow$ We have found the 4-th smallest element, so the 7-th smallest is in [10 13 8 11]

Randpartition(5, 8) $\rightarrow$ Again pick $A[5] = 10$ // first parameter of function

Get 5 3 2 6 [8 10 13 11]

$r = 6$ $k = 2$

$[6 \quad 10 \quad 13 \quad 5 \quad 8 \quad 3 \quad 2 \quad 11]$ $\quad n=8 \quad i=7$

$r \leftarrow$ RandPartition(1, 8) $\rightarrow$ Let's just "randomly" pick $x = A[1] = 6$

After $\Theta(n)$ work: $[5 \quad 3 \quad 2 \quad 6 \quad 10 \quad 13 \quad 8 \quad 11]$

$r = 4 = k$

We have found the 4-th smallest element, so the 7-th smallest is in $[10 \quad 13 \quad 8 \quad 11]$

Recursively call RandSelect(5, 8, 3) $\Leftarrow$ new $i = 7 - 4$

RandPartition(5, 8) $\rightarrow$ Again pick $A[5] = 10$ // first parameter of function

Get $5 \quad 3 \quad 2 \quad 6 \quad [8 \quad 10 \quad 13 \quad 11]$

$r = 6 \quad k = 2$

Recursively call RandSelect(7, 8, 1)

[6 10 13 5 8 3 2 11] $n=8$ $i=7$

$r \leftarrow$ RandPartition(1, 8) $\rightarrow$ Let's just "randomly" pick $x = A[1] = 6$

After $\Theta(n)$ work: [5 3 2 6 10 13 8 11]

$r = 4 = k$

Recursively call RandSelect(5, 8, 3) $\Leftarrow$ We have found the 4-th smallest element, so the 7-th smallest is in [10 13 8 11]

new $i = 7 - 4$

RandPartition(5, 8) $\rightarrow$ Again pick $A[5] = 10$ // first parameter of function

Get 5 3 2 6 [8 10 13 11]

$r = 6$ $k = 2$

Recursively call RandSelect(7, 8, 1)

RandPartition(7, 8) $\rightarrow$ $A[7] = 13$ $\rightarrow$ 5 3 2 6 8 10 [11 13]

$r = 8$

$k = 2$

trivial end

Analysis of RandSelect

$$T(n) = \underbrace{\Theta(n)}_{\text{RandPartition}} + T(f(n))$$

$f(n)$ = size of subarray
(either left or right of pivot)

"Perfect" pivot : splits array evenly. $T(n) = \Theta(n) + T(\frac{n}{2}) = \Theta(n)$

"Perfect" pivot : splits array evenly. $T(n) = \Theta(n) + T(\frac{n}{2}) = \Theta(n)$

"Just as perfect" : always split into constant fractions

even if you always recurse on "wrong" side. e.g. $T(n) = \Theta(n) + T(\frac{99n}{100}) = \Theta(n)$

"Perfect" pivot : splits array evenly. $T(n) = \Theta(n) + T(\frac{n}{2}) = \Theta(n)$

"Just as perfect" : always split into constant fractions

even if you always recurse on "wrong" side. e.g. $T(n) = \Theta(n) + T(\frac{99n}{100}) = \Theta(n)$

"Unlucky" : split 0 vs n-1 : $T(n) = \Theta(n) + T(n-1) = \Theta(n^2)$

"Perfect" pivot : splits array evenly. $T(n) = \Theta(n) + T(\frac{n}{2}) = \Theta(n)$

"Just as perfect" : always split into constant fractions

even if you always recurse on "wrong" side. e.g. $T(n) = \Theta(n) + T(\frac{99n}{100}) = \Theta(n)$

"Unlucky" : split 0 vs n-1 : $T(n) = \Theta(n) + T(n-1) = \Theta(n^2)$

"Just as unlucky" : split $\Theta(1)$ vs $n - \Theta(1)$: $T(n) = \Theta(n) + T(n-c)$

$$\approx n + (n-c) + (n-2c) + (n-3c) + \dots$$

$$\geq \frac{n}{2c} \cdot \frac{n}{2} = \Omega(n^2)$$

Expected time: call a split balanced if pivot ranks in $\left[\frac{n}{4} \dots \frac{3n}{4}\right]$

unbalanced otherwise

Expected time: call a split balanced if pivot ranks in $\left[\frac{n}{4} \dots \frac{3n}{4}\right]$
unbalanced otherwise

Worst case if balanced split: $T(n) \leq$?

Expected time: call a split balanced if pivot ranks in $[\frac{n}{4} \ldots \frac{3n}{4}]$
unbalanced otherwise

Worst case if balanced split: $T(n) \leq T(\frac{3n}{4}) + dn$

Worst case if unbalanced split: $T(n) \leq \ldots$

Expected time: call a split balanced if pivot ranks in $\left[\frac{n}{4} \dots \frac{3n}{4}\right]$
unbalanced otherwise

Worst case if balanced split: $T(n) \leq T\left(\frac{3n}{4}\right) + dn$

Worst case if unbalanced split: $T(n) \leq T(n-1) + dn < T(n) + dn$

Expected time: call a split balanced if pivot ranks in $\left[\frac{n}{4} \ldots \frac{3n}{4}\right]$
unbalanced otherwise

Worst case if balanced split: $T(n) \leq T\left(\frac{3n}{4}\right) + dn$

Worst case if unbalanced split: $T(n) \leq T(n-1) + dn < T(n) + dn$

Each split has a 50% chance of being balanced

Expected time: call a split balanced if pivot ranks in $\left[\frac{n}{4} \ldots \frac{3n}{4}\right]$

unbalanced otherwise

Worst case if balanced split: $T(n) \leq T\left(\frac{3n}{4}\right) + dn$

Worst case if unbalanced split: $T(n) \leq T(n-1) + dn < T(n) + dn$

Each split has a 50% chance of being balanced

$$T(n) \leq 0.5\underbrace{\left(T(n)+dn\right)}_{} + 0.5 \cdot \underbrace{\left(T\left(\frac{3n}{4}\right) + dn\right)}_{}$$

Expected time: call a split balanced if pivot ranks in $[\frac{n}{4} \dots \frac{3n}{4}]$

unbalanced otherwise

Worst case if balanced split: $T(n) \leq T(\frac{3n}{4}) + dn$

Worst case if unbalanced split: $T(n) \leq T(n-1) + dn < T(n) + dn$

Each split has a 50% chance of being balanced

$$T(n) \leq 0.5(T(n)+dn) + 0.5 \cdot (T(\tfrac{3n}{4}) + dn)$$

$$0.5\,T(n) \leq dn + 0.5 \cdot T(\tfrac{3n}{4})$$

Expected time: call a split balanced if pivot ranks in $[\frac{n}{4} \dots \frac{3n}{4}]$
unbalanced otherwise

Worst case if balanced split: $T(n) \leq T(\frac{3n}{4}) + dn$

Worst case if unbalanced split: $T(n) \leq T(n-1) + dn < T(n) + dn$

Each split has a 50% chance of being balanced

$$T(n) \leq 0.5(T(n)+dn) + 0.5 \cdot (T(\tfrac{3n}{4}) + dn)$$

$$0.5\,T(n) \leq dn + 0.5 \cdot T(\tfrac{3n}{4})$$

$$T(n) \leq T(\tfrac{3n}{4}) + 2dn = \Theta(n)$$

Expected time: call a split balanced if pivot ranks in $[\frac{n}{4} \dots \frac{3n}{4}]$
unbalanced otherwise

Worst case if balanced split: $T(n) \leq T(\frac{3n}{4}) + dn$

Worst case if unbalanced split: $T(n) \leq T(n-1) + dn < T(n) + dn$

Each split has a 50% chance of being balanced

$$T(n) \leq 0.5(T(n)+dn) + 0.5 \cdot (T(\frac{3n}{4}) + dn)$$

$$0.5\,T(n) \leq dn + 0.5 \cdot T(\frac{3n}{4})$$

$$T(n) \leq T(\frac{3n}{4}) + 2dn = \Theta(n)$$

$$2dn \cdot \frac{1}{1-3/4} = 8dn$$

Define $X_k \begin{cases} 1 & \text{if RandPartition gives } k \text{ vs } n-k-1 \text{ split.} \\ 0 & \text{otherwise} \end{cases}$

Define $X_k \begin{cases} 1 & \text{if RandPartition gives } k \text{ vs } n-k-1 \text{ split.} \\ 0 & \text{otherwise} \end{cases}$

always assuming worst scenario

$$T(n) \leqslant \Theta(n) + \text{one of}: \left.\begin{cases} T(\max\{0, n-1\}) \\ T(\max\{1, n-2\}) \\ T(\max\{2, n-3\}) \\ \vdots \\ T(\max\{n-1, 0\}) \end{cases}\right\} n \text{ possible outcomes}$$

$$\text{Define } X_k \begin{cases} 1 & \text{if RandPartition gives } k \text{ vs } n-k-1 \text{ split.} \\ 0 & \text{otherwise} \end{cases}$$

always assuming worst scenario

$$T(n) \leq \Theta(n) + \text{one of :} \underbrace{\begin{cases} T(\max\{0, n-1\}) \\ T(\max\{1, n-2\}) \\ T(\max\{2, n-3\}) \\ \vdots \\ T(\max\{n-1, 0\}) \end{cases}}_{\Theta(n) + \sum_{k=0}^{n-1} X_k \cdot T(\max\{k, n-k-1\})}$$

$$E\left[T(n)\right] \leq E\left[\Theta(n) + \sum_{k=0}^{n-1} X_k \cdot T\left(\max\{k, n-k-1\}\right)\right]$$

$$E\left[T(n)\right] \leq E\left[\Theta(n) + \sum_{k=0}^{n-1} X_k \cdot T(\max\{k, n-k-1\})\right]$$

$$= E\left[\Theta(n)\right] + \sum_{k=0}^{n-1} E\left[X_k \cdot T(\max\{k, n-k-1\})\right]$$

why?

$$E\left[T(n)\right] \leq E\left[\Theta(n) + \sum_{k=0}^{n-1} X_k \cdot T(\max\{k, n-k-1\})\right]$$

$$= E\left[\Theta(n)\right] + \sum_{k=0}^{n-1} E\left[X_k \cdot T(\max\{k, n-k-1\})\right]$$ by linearity of expectation

$$E[T(n)] \leq E\left[\Theta(n) + \sum_{k=0}^{n-1} X_k \cdot T(\max\{k, n-k-1\})\right]$$

$$= E[\Theta(n)] + \sum_{k=0}^{n-1} E[X_k \cdot T(\max\{k, n-k-1\})]$$ by linearity of expectation

$$= \Theta(n) + \sum E[X_k] \cdot E[T(\max\{k, n-k-1\})]$$ why?

$$E\left[T(n)\right] \leq E\left[\Theta(n) + \sum_{k=0}^{n-1} X_k \cdot T(\max\{k, n-k-1\})\right]$$

$$= E\left[\Theta(n)\right] + \sum_{k=0}^{n-1} E\left[X_k \cdot T(\max\{k, n-k-1\})\right]$$ by linearity of expectation

$$= \Theta(n) + \sum E[X_k] \cdot E\left[T(\max\{k, n-k-1\})\right]$$ by independence

$$E\big[T(n)\big] \leq E\left[\Theta(n) + \sum_{k=0}^{n-1} X_k \cdot T(\max\{k, n-k-1\})\right]$$

$$= E\big[\Theta(n)\big] + \sum_{k=0}^{n-1} E\big[X_k \cdot T(\max\{k, n-k-1\})\big]$$ by linearity of expectation

$$= \Theta(n) + \sum E[X_k] \cdot E\big[T(\max\{k, n-k-1\})\big]$$ by independence

$$= \Theta(n) + \sum_{k=0}^{n-1} \cdot \frac{1}{n} \cdot E\big[T(\max\{k, n-k-1\})\big]$$ by random choice

$$E\left[T(n)\right] \leq E\left[\Theta(n) + \sum_{k=0}^{n-1} X_k \cdot T\left(\max\{k, n-k-1\}\right)\right]$$

$$= E\left[\Theta(n)\right] + \sum_{k=0}^{n-1} E\left[X_k \cdot T\left(\max\{k, n-k-1\}\right)\right]$$

by linearity of expectation

$$= \Theta(n) + \sum E\left[X_k\right] \cdot E\left[T\left(\max\{k, n-k-1\}\right)\right]$$

by independence

$$= \Theta(n) + \sum_{k=0}^{n-1} \cdot \frac{1}{n} \cdot E\left[T\left(\max\{k, n-k-1\}\right)\right]$$

by random choice

$$\leq \Theta(n) + \frac{2}{n} \sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} E\left[T(k)\right]$$

$$E\left[T(n)\right] \leq \Theta(n) + \frac{2}{n} \sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} E\left[T(k)\right]$$

$$E[T(n)] \leqslant \Theta(n) + \frac{2}{n}\sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} E[T(k)]$$

Guess $E[T(n)] \leqslant c \cdot n$

assume true for $k < n$

$$E[T(n)] \leq \Theta(n) + \frac{2}{n} \sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} \underbrace{E[T(k)]}$$

Guess $E[T(n)] \leq c \cdot n$

assume true for $k<n$

$$\leq \Theta(n) + \frac{2}{n} \sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} \underbrace{ck}$$

$$E[T(n)] \leq \Theta(n) + \frac{2}{n}\sum_{k=\lfloor\frac{n}{2}\rfloor}^{n-1} E[T(k)]$$

Guess $E[T(n)] \leq c \cdot n$

assume true for $k<n$

$$\leq \Theta(n) + \frac{2}{n}\sum_{k=\lfloor\frac{n}{2}\rfloor}^{n-1} ck = \Theta(n) + \frac{2c}{n}\sum_{k=\lfloor\frac{n}{2}\rfloor}^{n-1} k$$

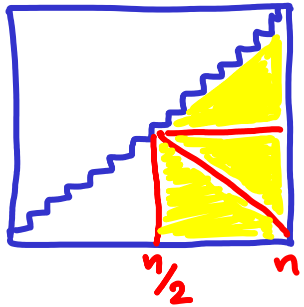
$$E[T(n)] \leq \Theta(n) + \frac{2}{n}\sum_{k=\lfloor\frac{n}{2}\rfloor}^{n-1} E[T(k)]$$

Guess $E[T(n)] \leq c \cdot n$

assume true for $k<n$

$$\leq \Theta(n) + \frac{2}{n}\sum_{k=\lfloor\frac{n}{2}\rfloor}^{n-1} ck = \Theta(n) + \frac{2c}{n}\sum_{k=\lfloor\frac{n}{2}\rfloor}^{n-1} k$$

$$\sum_{k=\lfloor\frac{n}{2}\rfloor}^{n-1} k \leq \frac{3}{8}n^2$$



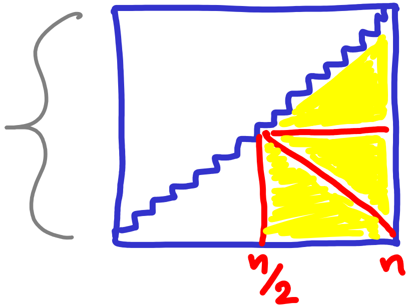
$$E[T(n)] \leq \Theta(n) + \frac{2}{n}\sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} E[T(k)]$$

Guess $E[T(n)] \leq c \cdot n$

assume true for $k<n$

$$\leq \Theta(n) + \frac{2}{n}\sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} ck = \Theta(n) + \frac{2c}{n}\sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} k$$

$$\sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} k \leq \frac{3}{8}n^2$$



$$\leq \Theta(n) + \frac{2c}{n}\cdot\frac{3}{8}n^2 = \Theta(n) + \frac{3cn}{4}$$

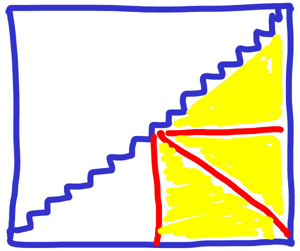
$$E[T(n)] \leq \Theta(n) + \frac{2}{n}\sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} E[T(k)]$$

Guess $E[T(n)] \leq c \cdot n$

assume true for $k<n$

$$\leq \Theta(n) + \frac{2}{n}\sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} ck = \Theta(n) + \frac{2c}{n}\sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} k$$

$$\sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} k \leq \frac{3}{8}n^2$$



$$\leq \Theta(n) + \frac{2c}{n} \cdot \frac{3}{8}n^2 = \Theta(n) + \frac{3cn}{4} = cn - \frac{1}{4}cn + dn$$

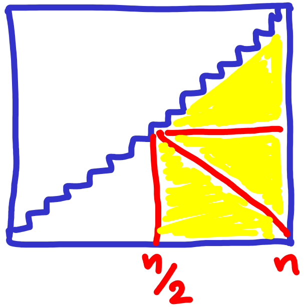
$$E[T(n)] \le \Theta(n) + \frac{2}{n}\sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} E[T(k)]$$

Guess $E[T(n)] \le c \cdot n$

↳ assume true for $k<n$

$$\le \Theta(n) + \frac{2}{n}\sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} ck = \Theta(n) + \frac{2c}{n}\sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} k$$

$$\sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} k \le \frac{3}{8}n^2$$



$$\le \Theta(n) + \frac{2c}{n}\cdot\frac{3}{8}n^2 = \underline{\Theta(n)} + \frac{3cn}{4} = cn - \frac{1}{4}cn + dn$$

$$= cn - \underbrace{\left(\frac{c}{4} - d\right)}_{\text{positive if } c>4d} n$$

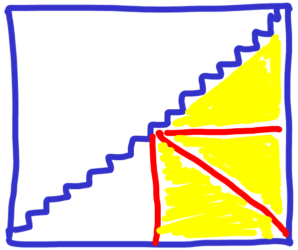
$$E[T(n)] \leq \Theta(n) + \frac{2}{n}\sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} E[T(k)]$$

Guess $E[T(n)] \leq c \cdot n$

↳ assume true for $k<n$

$$\leq \Theta(n) + \frac{2}{n}\sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} ck = \Theta(n) + \frac{2c}{n}\sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} k$$

$$\sum_{k=\lfloor \frac{n}{2} \rfloor}^{n-1} k \leq \frac{3}{8}n^2$$



$$\leq \Theta(n) + \frac{2c}{n}\cdot\frac{3}{8}n^2 = \underline{\Theta(n)} + \frac{3cn}{4} = cn - \frac{1}{4}cn + dn$$

$$= cn - \underbrace{\left(\frac{c}{4} - d\right)}_{\text{positive if } c>4d} n$$

$$E[T(n)] \leq 4dn$$