Lecture 2

Advance Algorithm Analysis (COMP 502 A)

Sharoon Nasim
sharoonnasim@fccollege.edu.pk

Office Hours:
Office: S-426 D
M,F 3-4 PM,
TR 9:30 AM - 11:30 PMz

Longest Common Subsequence

ababba
bababa

```
if x[i] == y[i]){

c[i][j] = 1+ c[i-1][j-1]

}
else{

c[i][j] =max(c[i-1][j], c[i][j-1])

}
```

abcd
ab, bc, cd, ac
bd, cd, ad

a b a b a

	0	a	b	a	b	b	a
0	0	0	0	0	0	0	0
b	0	0	1	1	1	1	1
a	0	1	1	2	2	2	2
b	0	1	2	2	3	3	3
a	0	1	2	3	3	3	4
b	0	1	2	3	4	4	4
a	0	1	2	3	4	4	5

Divide and Conquer

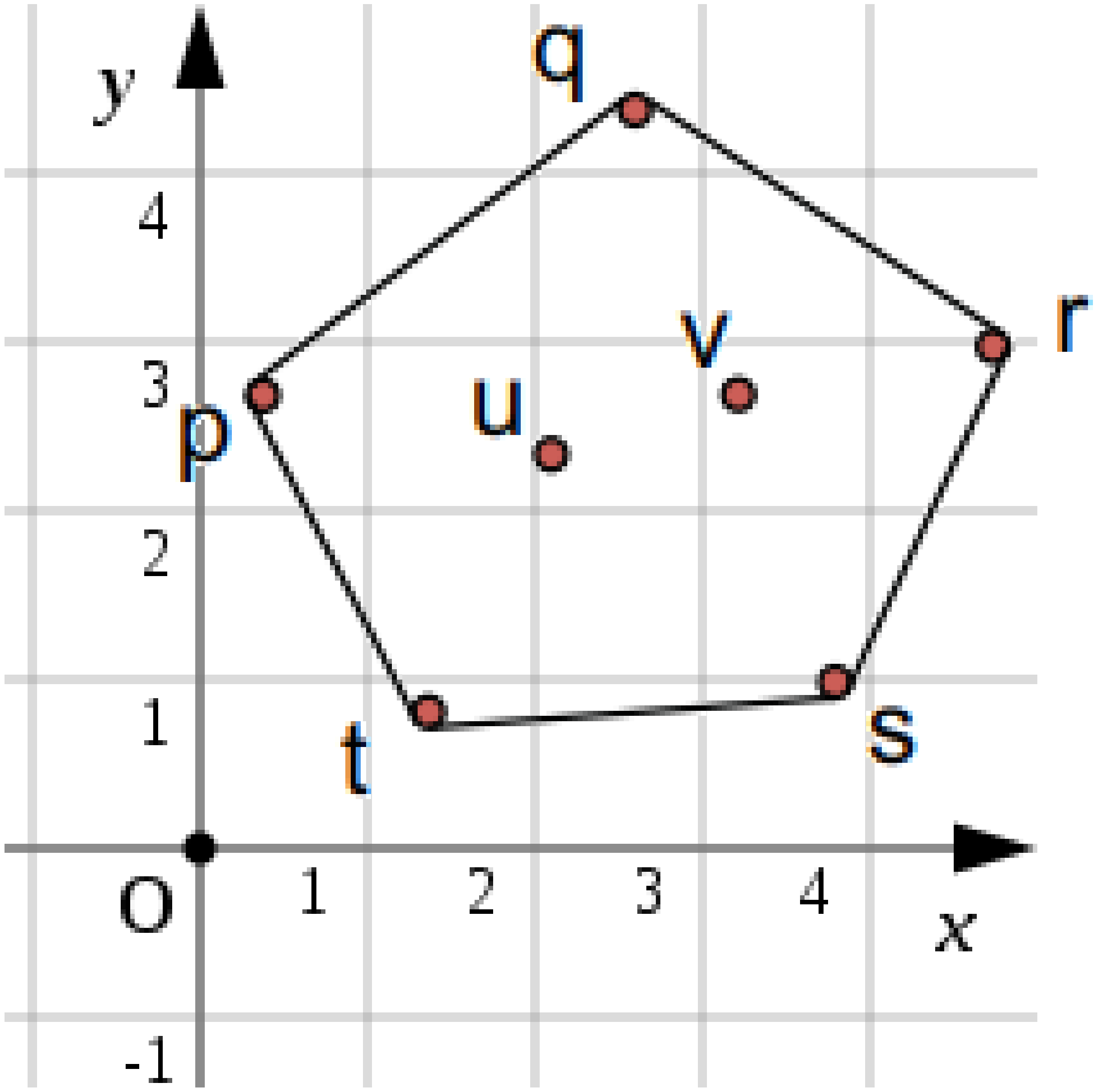
Paradigm

Given a problem of size n divide it into subproblems of size $\frac{n}{b}$, $a \geq 1$, $b > 1$. Solve each subproblem recursively. Combine solutions of subproblems to get overall solution.

$$T(n) = aT(\frac{n}{b}) + [\text{work for merge}]$$

Convex Hull

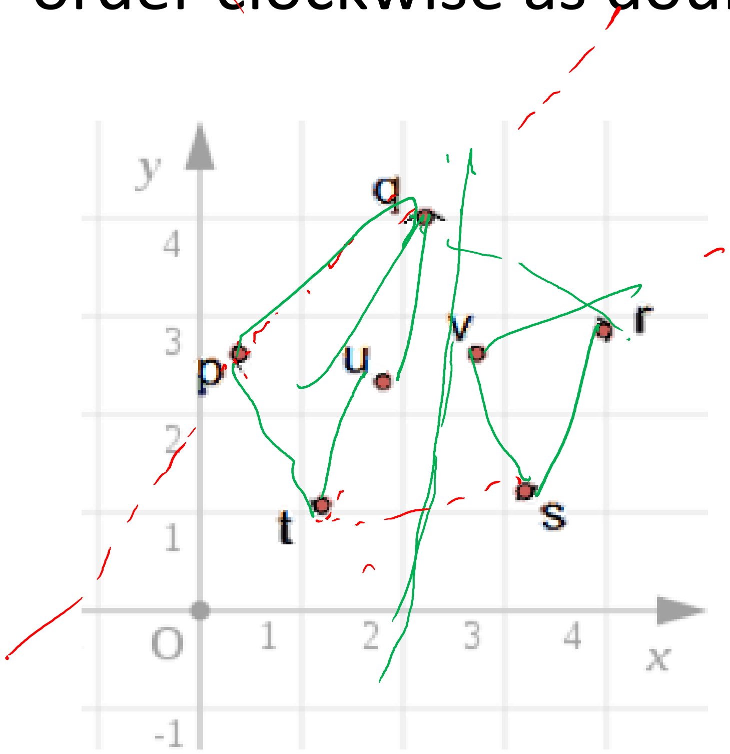
Given n points in plane
S = {(xi, yi) | i = 1, 2,...,n}
assume no two have same x coordinate, no two have same y coordinate, and no three in a line.

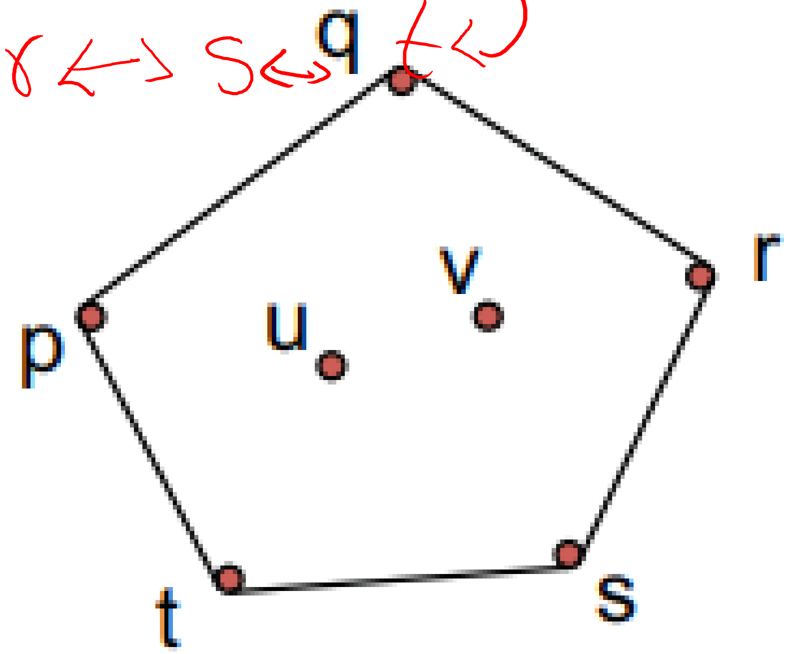


Problem Statement:

- Convex Hull (CH(S)): smallest polygon containing all points in S.
- CH(S) represented by the sequence of points on the boundary in order clockwise as doubly linked list.

p ⟷ q ⟷ r ⟷ s ⟷ t





Brute Force Approach to solve Convex Hull

Brute force for Convex Hull Test each line segment to see if it makes up an edge of the convex hull

- If the rest of the points are on one side of the segment, the segment is on the convex hull.
- else the segment is not.

O(n2) edges, O(n) tests ⇒ O(n3) complexity

Line of equation y = mx + c

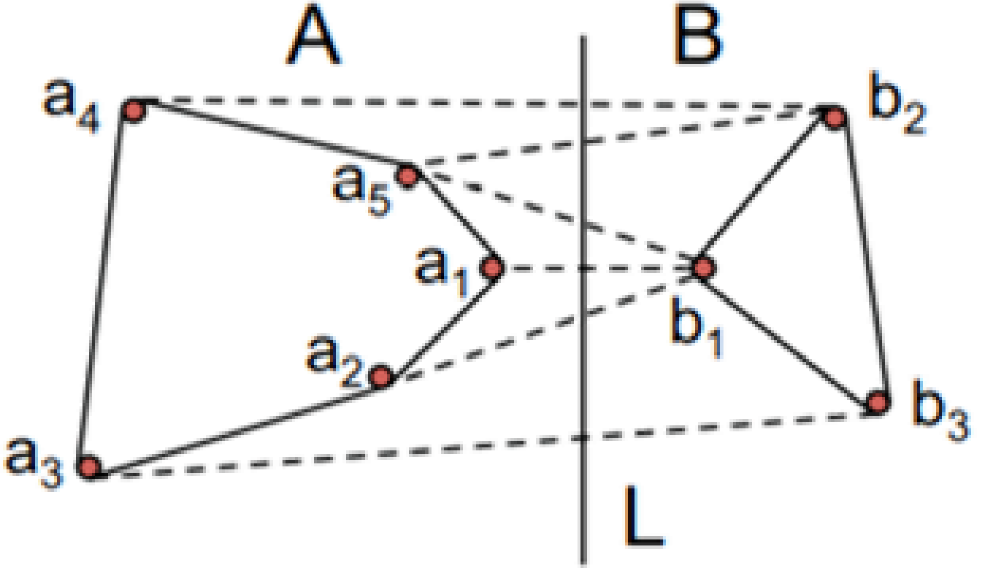
Divide and Conquer Convex Hull

Sort points by x coord (once and for all, O(n log n))

For input set S of points:

- Divide into left half A and right half B by x coords
- Compute CH(A) and CH(B)
- Combine CH's of two halves (merge step)

How to Merge?



Finding Tangents

Assume a_i maximizes x within $CH(A)$ $(a_1, a_2, \ldots, a_p)$. b_1 minimizes x within $CH(B)$ $(b_1, b_2, \ldots, b_q)$

L is the vertical line separating A and B. Define $y(i, j)$ as y-coordinate of intersection between L and segment (a_i, b_j).

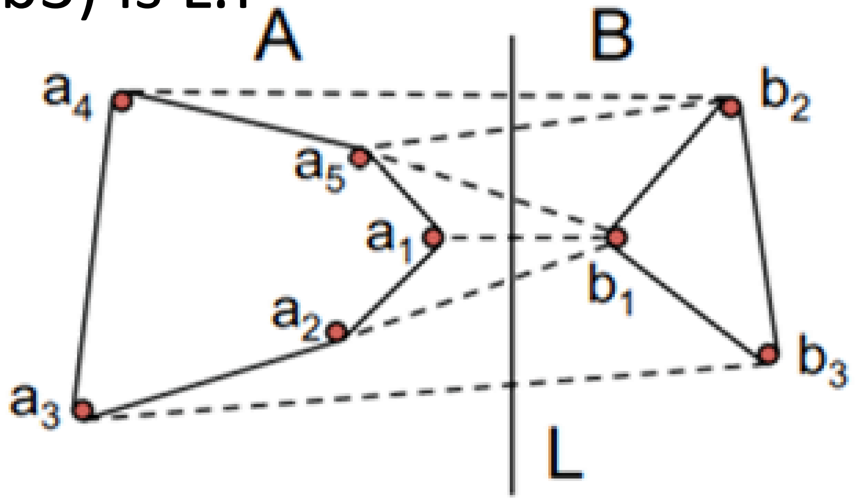
Claim: (a_i, b_j) is uppertangent iff it maximizes $y(i, j)$.

If $y(i, j)$ is not maximum, there will be points on both sides of (a_i, b_j) and it cannot be a tangent.

Algorithm: Obvious $O(n^2)$ algorithm looks at all a_i, b_j pairs. $T(n) = 2T(n/2) + \Theta(n^2) = \Theta(n^2)$.

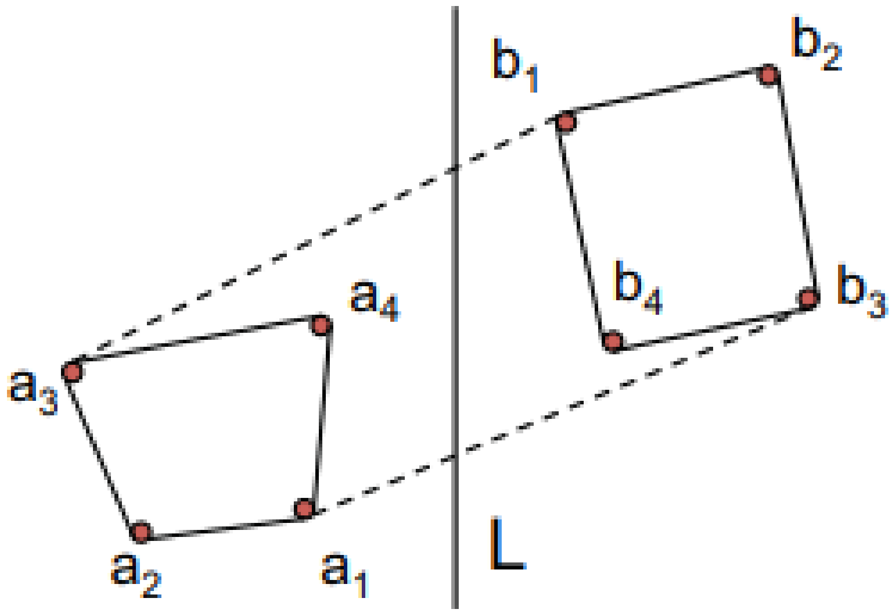
Convex Hull : Divide and Conquer

- Find upper tangent (ai, bj). In example, (a4, b2) is U.T.
- Find lower tangent (ak, bm). In example, (a3, b3) is L.T
- Cut and paste in time Θ(n).



First link a_i to b_j, go down b ilst till you see b_m and link b_m to a_k, continue along the a list until you return to a_i. In the example, this gives (a_4, b_2, b_3, a_3).

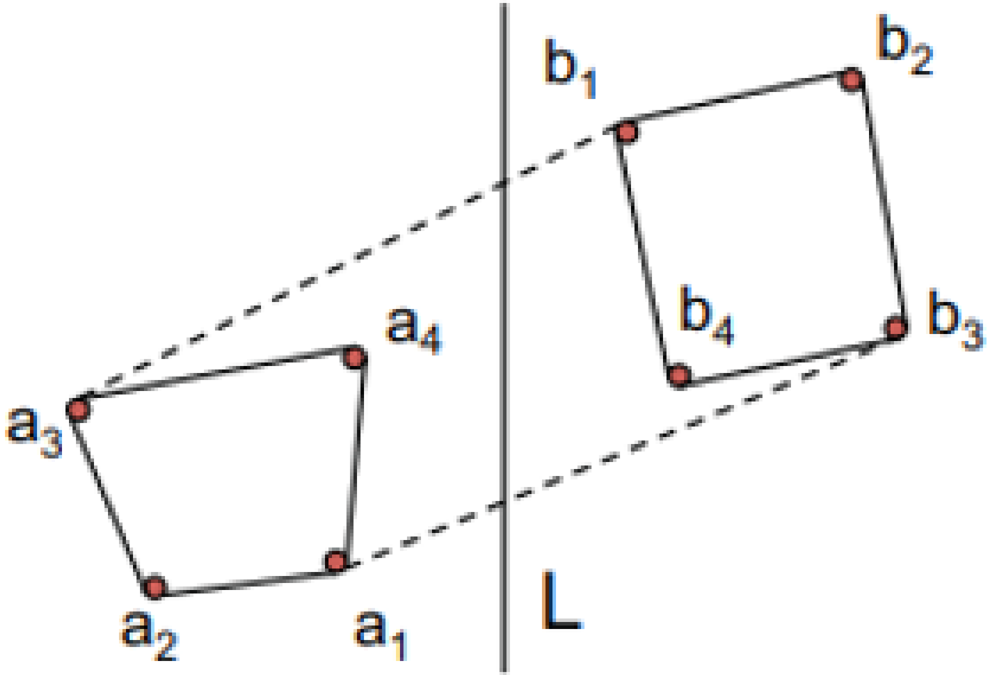
Example



a_3, b_1 is upper tangent. $a_4 > a_3$, $b_2 > b_1$ in terms of Y coordinates.
a_1, b_3 is lower tangent, $a_2 < a_1$, $b_4 < b_3$ in terms of Y coordinates.

a_i, b_j is an upper tangent. Does not mean that a_i or b_j is the highest point. Similarly, for lower tangent.

Convex Hull



i = 1
j = 1
while $(y(i, j+1) > y(i, j)$ or $y(i-1, j) > y(i, j))$
    if $(y(i, j+1) > y(i, j))$ ▷ move right finger clockwise
        $j = j + 1 \pmod q$
    else
        $i = i - 1 \pmod p$ ▷ move left finger anti-clockwise
    return (a_i, b_j) as upper tangent

Similarly for lower tangent.

$$T(n) = 2T(\frac{n}{2}) + \Theta(n) = \Theta(n \log n)$$

Solving Recurrence Relation

$$T(n) = 2T(\frac{n}{2}) + \Theta(n) = \Theta(n \log n)$$

Master Theorem

Master theorem[2] *If* $T(n) = aT(\lceil n/b \rceil) + O(n^d)$ *for some constants* $a > 0$, $b > 1$, *and* $d \geq 0$,

$a=2 \quad b=2 \quad d=1$

ratio a/b^d. Finding the sum of such a series in big-O notation is easy (Exercise 0.2), and comes down to three cases.

$\frac{2}{2^1} = 1$

1. *The ratio is less than* 1.

   Then the series is decreasing, and its sum is just given by its first term, $O(n^d)$.

2. *The ratio is greater than* 1.

   The series is increasing and its sum is given by its last term, $O(n^{\log_b a})$:

$$n^d \left(\frac{a}{b^d}\right)^{\log_b n} = n^d \left(\frac{a^{\log_b n}}{(b^{\log_b n})^d}\right) = a^{\log_b n} = a^{(\log_a n)(\log_b a)} = n^{\log_b a}.$$

$O(n\log(n))$

$$a^{\log_b n} = b^{\log_b a\,(\log_b n)} = b^{(\log_b n)\log_b a} = n^{\log_b a}$$

3. *The ratio is exactly* 1.

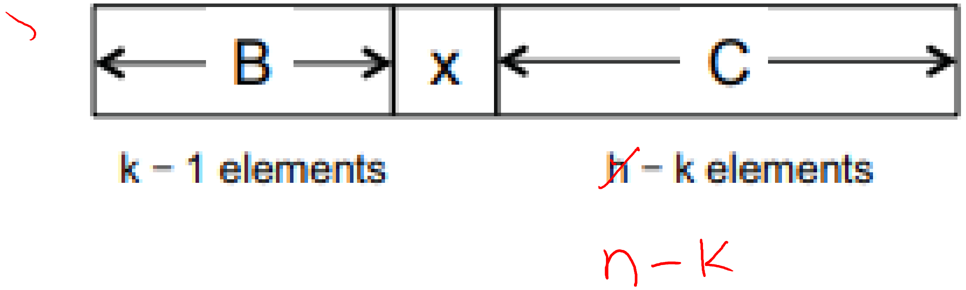
   In this case all $O(\log n)$ terms of the series are equal to $O(n^d)$. $= O(\log n\ (n^d))$

Median Finding

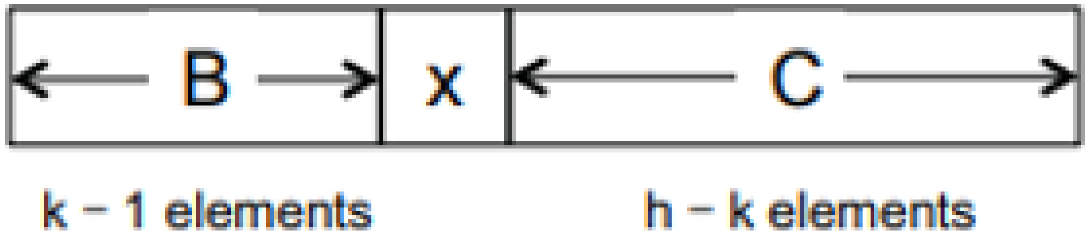
Given set of n numbers, define $rank(x)$ as number of numbers in the set that are $\le x$. Find element of rank $\lfloor \frac{n+1}{2} \rfloor$ (lower median) and $\lceil \frac{n+1}{2} \rceil$ (upper median).

Clearly, sorting works in time $\Theta(n \log n)$.

Can we do better?



Select(S,i):




```
    pick x ϵ S
    To compute k = rank(x)
    B = {y ϵ S | y < x}
    C = {y ϵ S | y > x}

    if k == i: return x
    else if  k> i: return Select(B,i)
    else if k < i: return (Select(C, i-k)
```

Picking x Cleverly

Need to pick x so $rank(x)$ is not extreme.

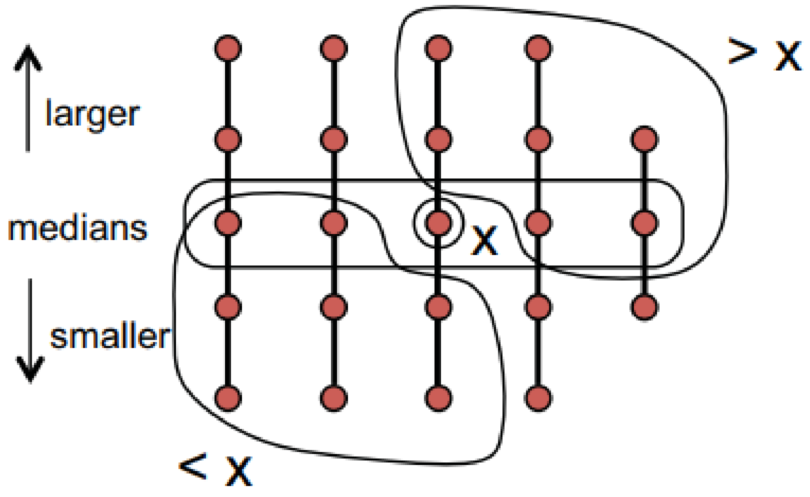
- Arrange S into columns of size 5 ($\lceil \frac{n}{5} \rceil$ cols)
- Sort each column (bigger elements on top) (linear time)
- Find "median of medians" as x

$\frac{n}{5}$ $\lceil \frac{23}{5} \rceil = 5$

$5/2 = 2 \times 3 = 6$

$\frac{\lceil \frac{n}{5} \rceil}{2}$

$\lceil \frac{n}{10} \rceil \times 3$



How many elements are guaranteed to be $> x$?

Half of the $\lceil \frac{n}{5} \rceil$ groups contribute at least 3 elements $> x$ except for 1 group with less than 5 elements and 1 group that contains x.

At lease $3(\lceil \frac{n}{10} \rceil - 2)$ elements are $> x$, and at least $3(\lceil \frac{n}{10} \rceil - 2)$ elements are $< x$

Recurrence:

$n - \left(\frac{3n}{10} - 2\right) = \frac{10n - 3n}{10} + 6 = \frac{7n}{10} + 6$

$$T(n) = \begin{cases} O(1) & n \leq 140 \\ T(T\left(\left\lceil \frac{n}{5} \right\rceil\right) + T\left(\frac{7n}{10} + 6\right) + \theta(n)) & n > 140 \end{cases}$$

Overall Time $= \theta(n)$ as n/5 and 7n/10 is also less than n