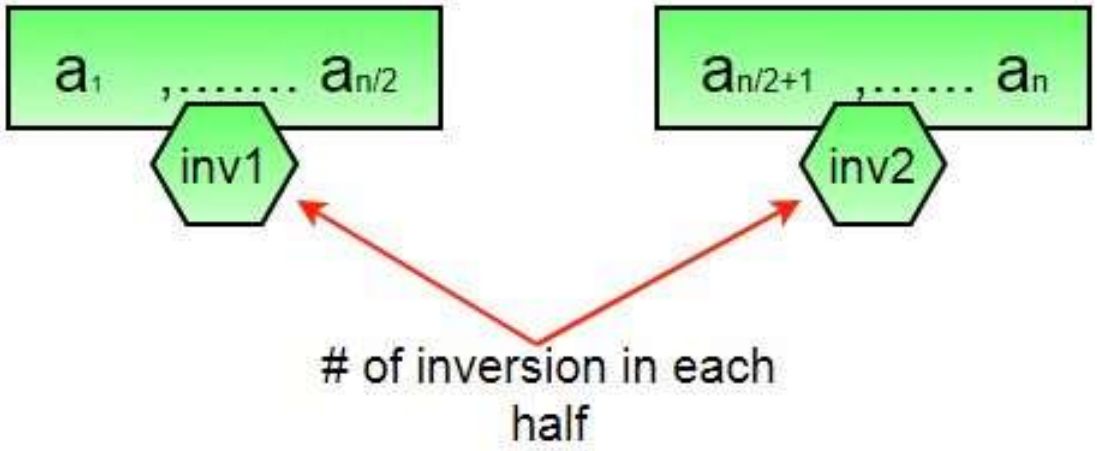
Lecture 16

Brute Force Algorithms & their Analysis: Finding Inversion in an Array, and Closest-Pair & Convex-Hull Problems.

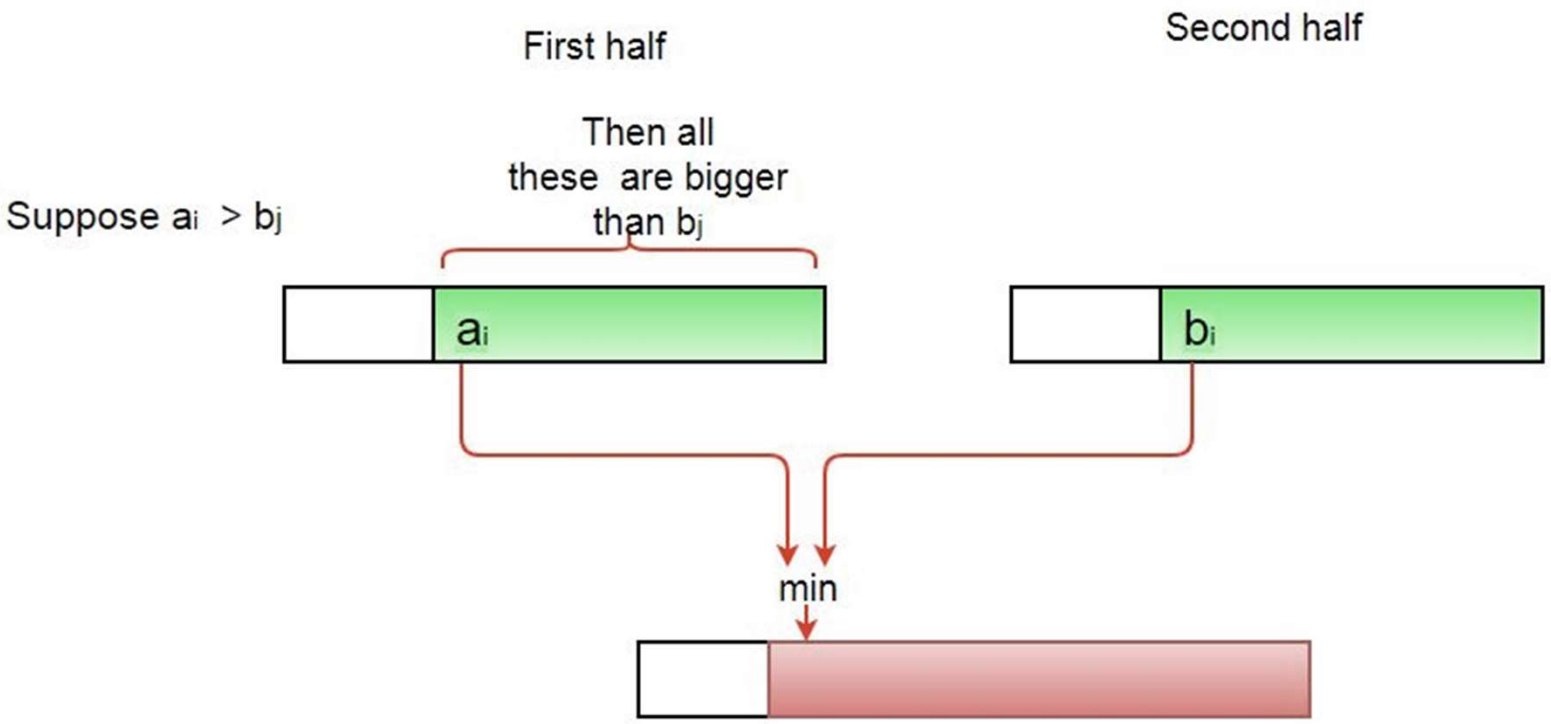
Inversion Count in Array

- **Inversion Count** for an array indicates – how far (or close) the array is from being sorted.
  - If the array is already sorted, then the inversion count is 0,
  - if the array is sorted in reverse order, the inversion count is the maximum.
- Given an array **a[]**. The task is to find the inversion count of **a[]**. Where two elements a[i] and a[j] form an inversion if a[i] > a[j] and i < j

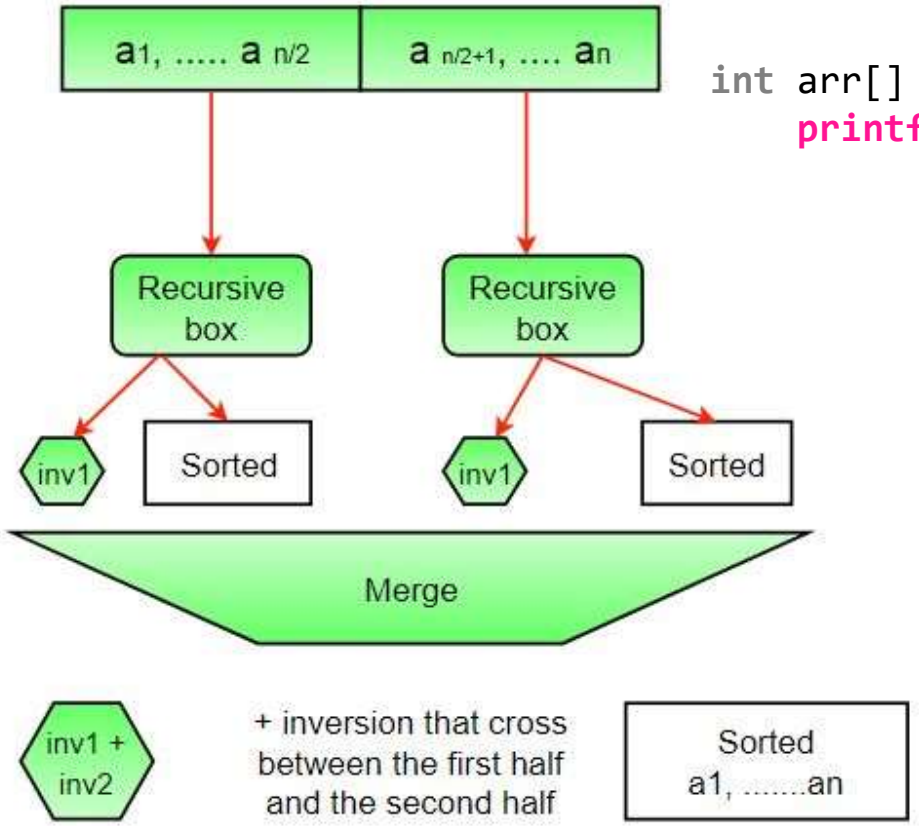
Concept of Inversion



Concept of Inversion



Concept of Inversion




```
int arr[] = { 1, 20, 6, 4, 5 };
    printf(" Number of inversions are %d \n",
           mergeSort(arr, 5));
```

Time Complexity:
O(n log n), The algorithm used is divide and conquer i.e. merge sort whose complexity is O(n log n).
Auxiliary Space:
O(n), Temporary array.

Inversions in Array: Steps {Self Execution}

```
GetInvCount(ar, n):

      inv_count = 0
    for i in range(n):
        for j in range(i + 1, n):
            if (ar[i] > ar[j]):
                inv_count += 1

      return inv_count
```

```
array = [1, 20, 6, 4, 5]
n = len(array)
print("Number of inversions are",
      GetInvCount(array, n))
```

OUTPUT:

Number of inversions are 5

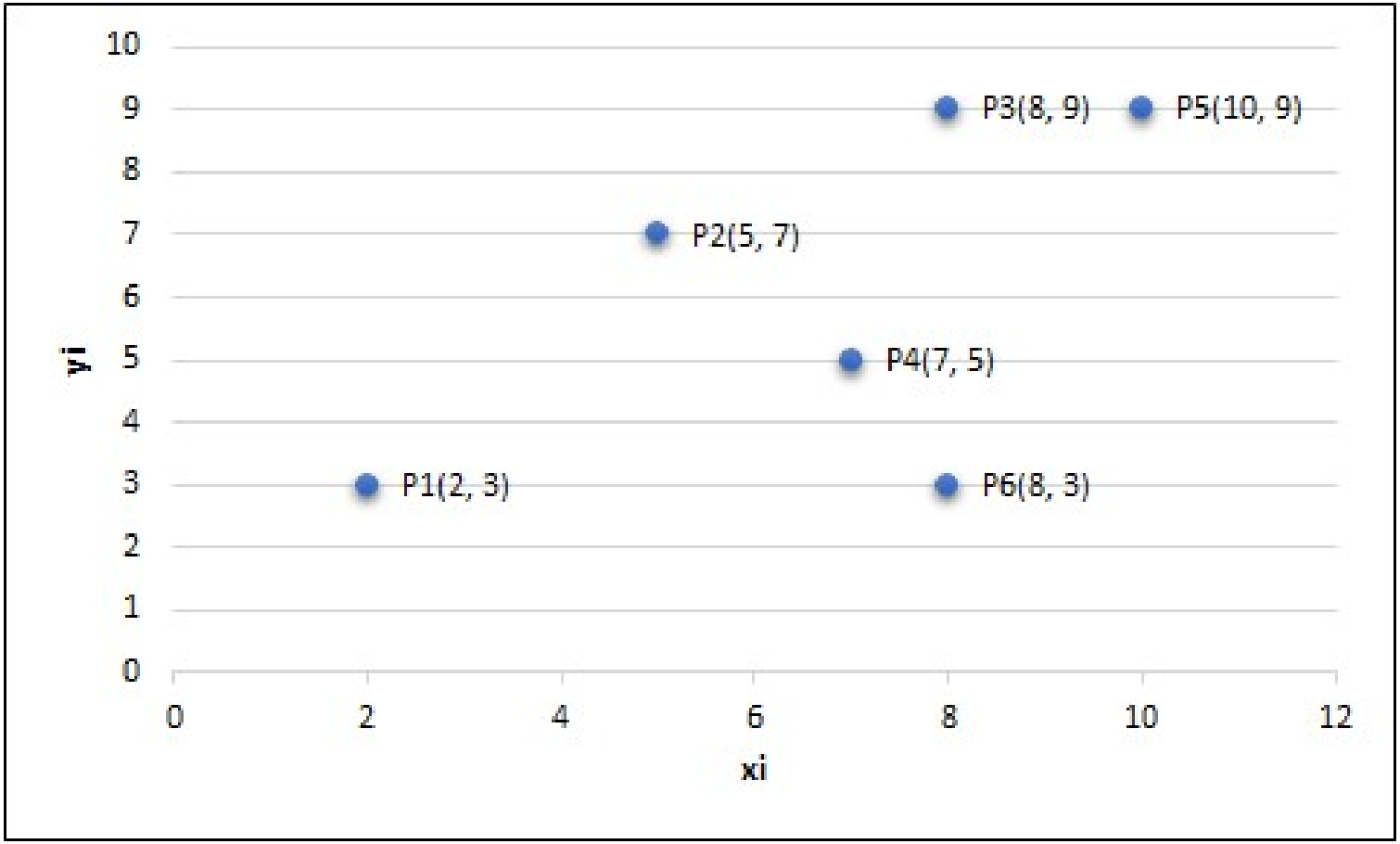
- **Time Complexity:** $O(n^2)$, Two nested loops are needed to traverse the array from start to end.
- **Auxiliary Space:** O(1), No extra space is required.

Inversion Count in Array: Examples

- ***Input:*** *arr[] = {8, 4, 2, 1}*
  Output: *6*
  Explanation: *Given array has six inversions:*
  - *(8, 4), (4, 2), (8, 2), (8, 1), (4, 1), (2, 1).*

- ***Input:*** *arr[] = {1, 20, 6, 4, 5}*
  Output: *5*
  Explanation: *Given array has five inversions:*
  - *(20, 6), (20, 4), (20, 5), (6, 4), (6, 5).*

Brute Force: Closest Pair



Brute Force: Closest Pair Problem

› Find distance between points

P1P2	P2P3	P3P4	P4P5	P5P6
P1P3	P2P4	P3P5	P4P6	
P1P4	P2P5	P3P6		
P1P5	P2P6			
P1P6				

› Formula to use: This is the basic distance formula

– $d(p_i, p_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$

– The least distance shows the closest pair

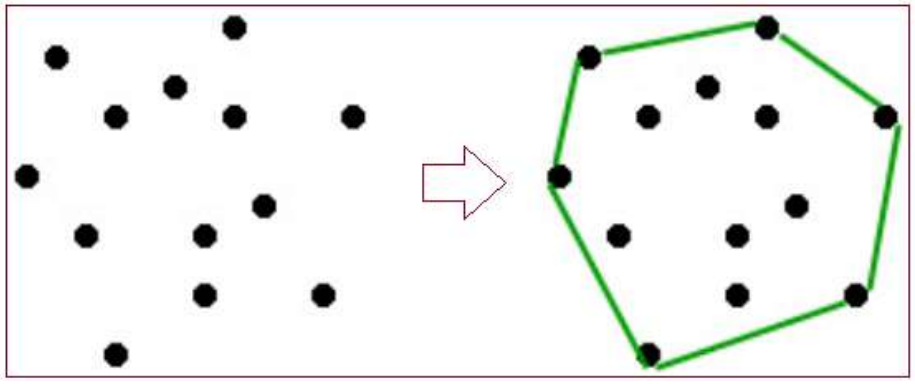
Brute Force: Closest Pair Algorithm

- $\boldsymbol{Algorithm\ BF - ClosestPair(p)}$
- dmin $= \infty$
- $FOR\ i = 1\ to\ n - 1$
  - $FOR\ j = i + 1\ to\ n$
    - $d = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$
    - $IF\ d < dmin\ THEN$
      - $dmin = d$
      - $index1, = i$
      - $index2 = j$
- $RETURN\ indexi, index2$

- $\boldsymbol{Time\ Complexity}$
  - $C(n) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} 2$
  - $= 2 \sum_{i=1}^{n-1} (n - i - 1 + 1)$
  - $= 2 \sum_{i=1}^{n-1} (n - i)$
  - $= 2(n-1) + (n-2) \ldots + 1)$
  - $= 2.\frac{n(n-1)}{2} = n(n-1) = n^2 - n$
  - $\Theta(n^2) \cong C(n)$

Convex-Hull Problem : Divide & Conquer

- Convex-Hull is the smallest convex polygon that contain all the points in the plane.
  - Given a set of X of point $P_1(x_1, y_1)$, $P_2(x_2, y_2)$ …., $P_n(x_n, y_n)$ in plane. We want to find convex hull.
  - Divide and Conquer algorithm take $O(n \log n)$ time to compute convex hull in clockwise order.
  - All the points are inside the polygon after connecting all outer points and called as Convex –Hull.
  - Input points must be pre-sorted by x-coordinates



Divide & Conquer: Convex Hull Steps

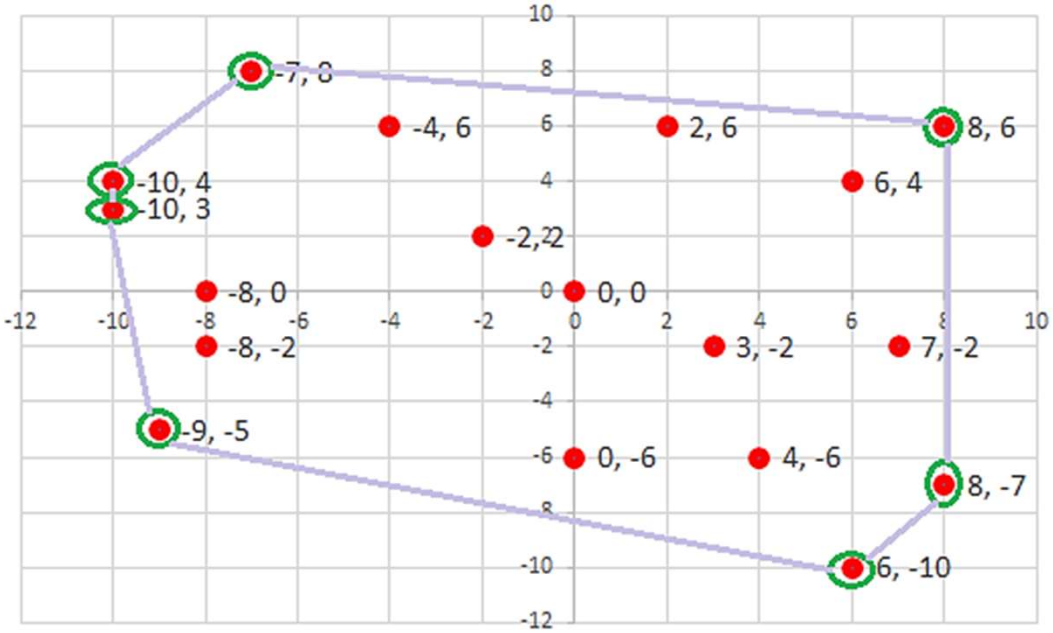
1. Partition X into half X_1 and X_2 according to X coordinates.
2. If X1 is Empty, THEN Upper Half is simply line with end points P_1 and P_2
3. If X is not Empty, THEN Find P_{max} in X_1 which is fastest from linear P_1P_n
4. If the TIE in P_{max} THEN point the maximum angle P_{max}, P_1, P_2 can be
5. Now, Algorithm identifies all point if X_1 that are left of line P_1P_{max} – goto Step2

Divide and conquer : *Convex Hulls*

1. Divide the n points into two halves.
2. Find convex hull of each subset.
3. Combine the two hulls into overall convex hull.

Convex-Hull Example

- **Input** – Set of points: {(-7,8), (-4,6), (2,6), (6,4), (8,6), (7,-2), (4,-6), (8,-7),(0,0), (3,-2),(6,-10),(0,-6),(-9,-5),(-8,-2),(-8,0),(-10,3),(-2,2),(-10,4)}
- **Output** – Boundary points of convex hull are –
  - (-9, -5) (6, -10) (8, -7) (8, 6) (-7, 8) (-10, 4) (-10, 3)



https://www.tutorialspoint.com/online_cpp_compiler.php

Thank You!!!

Have a good day