

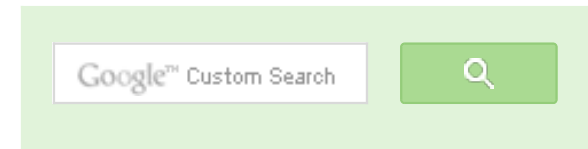
## Given n line segments, find if any two segments intersect

We have discussed the problem to detect if **two given line segments intersect or not**. In this post, we extend the problem. Here we are given n line segments and we need to find out if any two line segments intersect or not.

**Naive Algorithm** A naive solution to solve this problem is to check every pair of lines and check if the pair intersects or not. **We can check two line segments in  $O(1)$  time**. Therefore, this approach takes  $O(n^2)$ .

**Sweep Line Algorithm:** We can solve this problem in  **$O(n \log n)$**  time using Sweep Line Algorithm. The algorithm first sorts the end points along the x axis from left to right, then it passes a vertical line through all points from left to right and checks for intersections. Following are detailed steps.

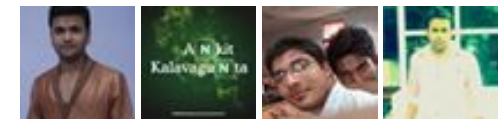
- 1) Let there be n given lines. There must be 2n end points to represent the n lines. Sort all points according to x coordinates. While sorting maintain a flag to indicate whether this point is left point of its line or right point.
- 2) Start from the leftmost point. Do following for every point
  - .....a) If the current point is a left point of its line segment, check for intersection of its line segment with the segments just above and below it. And add its line to *active* line segments (line segments for which left end point is seen, but right end point is not seen yet). Note that we consider only those neighbors which are still active.
  - ....b) If the current point is a right point, remove its line segment from active list and check whether its two active neighbors (points just above and below) intersect with each other.



GeeksforGeeks



53,525 people like [GeeksforGeeks](#).



[Interview Experiences](#)

[Advanced Data Structures](#)

[Dynamic Programming](#)

[Greedy Algorithms](#)

[Backtracking](#)

[Pattern Searching](#)

[Divide & Conquer](#)

[Mathematical Algorithms](#)

[Recursion](#)

[Geometric Algorithms](#)

The step 2 is like passing a vertical line from all points starting from the leftmost point to the rightmost point. That is why this algorithm is called Sweep Line Algorithm. The Sweep Line technique is useful in many other geometric algorithms like [calculating the 2D Voronoi diagram](#)

### What data structures should be used for efficient implementation?

In step 2, we need to store all active line segments. We need to do following operations efficiently:

- Insert a new line segment
- Delete a line segment
- Find predecessor and successor according to y coordinate values

The obvious choice for above operations is Self-Balancing Binary Search Tree like AVL Tree, Red Black Tree. With a Self-Balancing BST, we can do all of the above operations in  $O(\log n)$  time.

Also, in step 1, instead of sorting, we can use min heap data structure. Building a min heap takes  $O(n)$  time and every extract min operation takes  $O(\log n)$  time (See [this](#)).

### PseudoCode:

The following pseudocode doesn't use heap. It simply sort the array.

**sweepLineIntersection(Points[0..2n-1]):**

- Sort Points[] from left to right (according to x coordinate)
- Create an empty Self-Balancing BST T.

// Process all 2n points

- for  $i = 0$  to  $2n-1$

// If this point is left end of its line

if (Points[i].isLeft)

T.insert(Points[i].line()) // Insert into the tree

// Check if this points intersects with its predecessor and successor

if ( doIntersect(Points[i].line(), T.pred(Points[i].line()) )

return true

if ( doIntersect(Points[i].line(), T.succ(Points[i].line()) )

return true

## HP Chromebook 11

 [google.com/chromebook](https://google.com/chromebook)

Everything you need in one laptop.  
Made with Google. Learn more.



## Popular Posts

[All permutations of a given string](#)

[Memory Layout of C Programs](#)

[Understanding "extern" keyword in C](#)

[Median of two sorted arrays](#)

[Tree traversal without recursion and without stack!](#)

[Structure Member Alignment, Padding and Data Packing](#)

[Intersection point of two Linked Lists](#)

[Lowest Common Ancestor in a BST.](#)

[Check if a binary tree is BST or not](#)

[Sorted Linked List to Balanced BST](#)

```

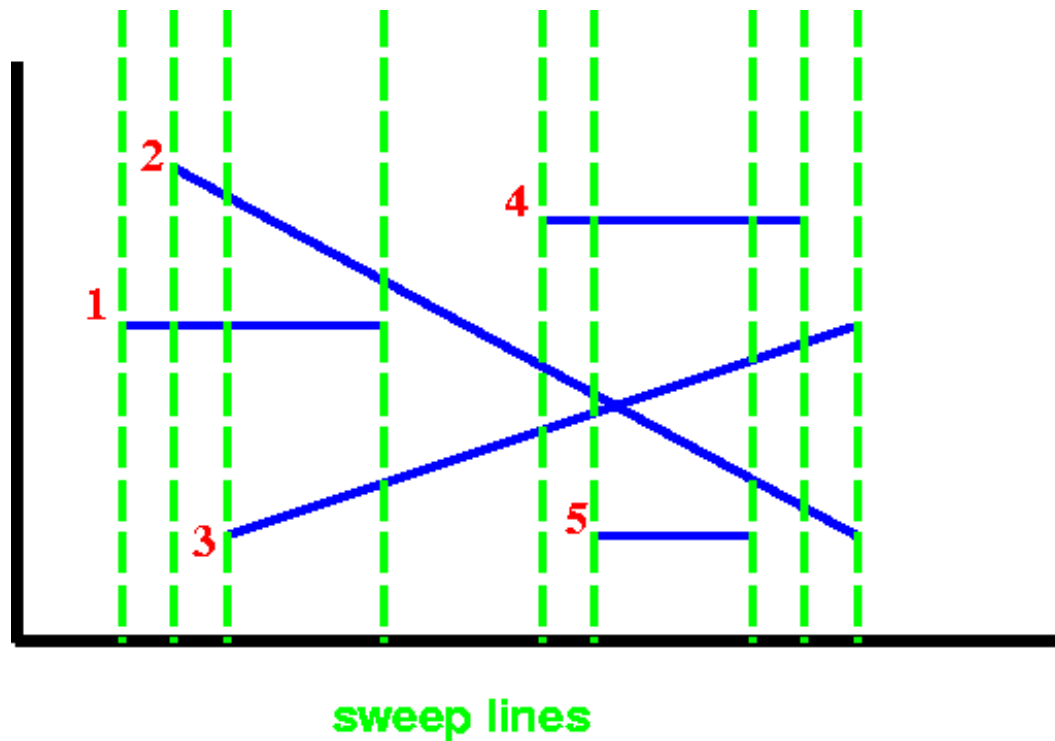
else // If it's a right end of its line
    // Check if its predecessor and successor intersect with each other
    if ( doIntersect(T.pred(Points[i].line()), T.succ(Points[i].line())))
        return true
    T.delete(Points[i].line()) // Delete from tree

```

4. return False

#### Example:

Let us consider the following example taken from [here](#). There are 5 line segments 1, 2, 3, 4 and 5. The dotted green lines show sweep lines.



Following are steps followed by the algorithm. All points from left to right are processed one by one. We maintain a self-balancing binary search tree.

*Left end point of line segment 1 is processed:* 1 is inserted into the Tree. The tree contains 1. No intersection.

UP TO  
**60%** OFF  
**INDUSTRIAL  
RUSTIC  
DESIGNS**

**SHOP THIS LOOK >**

**Dot&Bo**

## Recent Comments

Abhi You live US or India?

[Google \(Mountain View\) interview](#) · 9 minutes ago

[Aman](#) Hi, Why arent we checking for conditions...

[Write a C program to Delete a Tree.](#) · 49 minutes ago

kzs please provide solution for the problem...

[Backtracking | Set 2 \(Rat in a Maze\)](#) · 52 minutes ago

[Sanjay Agarwal](#) bool

tree::Root\_to\_leaf\_path\_given\_sum(tree...

[Root to leaf path sum equal to a given number](#) · 1 hour ago

[GOPI GOPINATH @admin](#) Highlight this sentence "We can easily...

[Count trailing zeroes in factorial of a number](#) · 1 hour ago

[newCoder3006](#) If the array contains negative numbers also. We...

[Find subarray with given sum](#) · 1 hour ago

AdChoices 

[► Segments](#)

[► Planes Intersect](#)

[► Java Algorithm](#)

AdChoices 

*Left end point of line segment 2 is processed:* Intersection of 1 and 2 is checked. 2 is inserted into the Tree. No intersection. The tree contains 1, 2.

*Left end point of line segment 3 is processed:* Intersection of 3 with 1 is checked. No intersection. 3 is inserted into the Tree. The tree contains 2, 1, 3.

*Right end point of line segment 1 is processed:* 1 is deleted from the Tree. Intersection of 2 and 3 is checked. Intersection of 2 and 3 is reported. The tree contains 2, 3. Note that **the above pseudocode returns at this point**. We can continue from here to report all intersection points.

*Left end point of line segment 4 is processed:* Intersections of line 4 with lines 2 and 3 are checked. No intersection. 4 is inserted into the Tree. The tree contains 2, 4, 3.

*Left end point of line segment 5 is processed:* Intersection of 5 with 3 is checked. No intersection. 4 is inserted into the Tree. The tree contains 2, 4, 3, 5.

*Right end point of line segment 5 is processed:* 5 is deleted from the Tree. The tree contains 2, 4, 3.

*Right end point of line segment 4 is processed:* 4 is deleted from the Tree. The tree contains 2, 4, 3. Intersection of 2 with 3 is checked. Intersection of 2 with 3 is reported. The tree contains 2, 3. Note that the intersection of 2 and 3 is reported again. We can add some logic to check for duplicates.

*Right end point of line segment 2 and 3 are processed:* Both are deleted from tree and tree becomes empty.

**Time Complexity:** The first step is sorting which takes  $O(n \log n)$  time. The second step process  $2n$  points and for processing every point, it takes  $O(\log n)$  time. Therefore, overall time complexity is  $O(n \log n)$

### References:

<http://www.cs.uiuc.edu/~jeffe/teaching/373/notes/x06-sweepine.pdf>

<http://courses.csail.mit.edu/6.006/spring11/lectures/lec24.pdf>

<http://www.youtube.com/watch?v=dePDHVovJIE>

<http://www.eecs.wsu.edu/~cook/aa/lectures/l25/node10.html>

Please write comments if you find anything incorrect, or you want to share more information


about the topic discussed above.

# Informix Database

 [progress.com/Informix](https://progress.com/Informix)

JDBC Compliant w/ Major  
Databases Fully Functional Eval -  
Try Now!



- [► Solve Equation](#)
- [► Data Points](#)
- [► Geometry Tangent](#)
- AdChoices 
- [► Geometry Tangent](#)
- [► Polygon Solution](#)
- [► Geeks](#)

## Related Tpoics:

---

- [Backtracking | Set 8 \(Solving Cryptarithmic Puzzles\)](#)
- [Tail Recursion](#)
- [Find if two rectangles overlap](#)
- [Analysis of Algorithm | Set 4 \(Solving Recurrences\)](#)
- [Print all possible paths from top left to bottom right of a mXn matrix](#)
- [Generate all unique partitions of an integer](#)
- [Russian Peasant Multiplication](#)
- [Closest Pair of Points | O\(nlogn\) Implementation](#)



27

 Tweet

3



1

Writing code in comment? Please use [ideone.com](https://ideone.com) and share the link here.

16 Comments

GeeksforGeeks

Sort by Newest ▼



Join the discussion...



**Tejas Patel** • 2 months ago

Why do we need to check only two segments and not more?

^ | v •



**Savan Popat** • 6 months ago

In "Left end point of line segment 5 is processed" , it should be "5 inserted in to

^ | v •



**anonymoe** • 6 months ago

I am new here but Can we not do something like this

We use result from 1st line segment comparison with other line segments and  
to the left of line segment and 1 to the right based on the orientation properties

^ | v •



**Kartik** → anonymoe • 6 months ago

Divide and Conquer is good idea. But the question is, how to write the  
 $O(n)$ . For example, for closest pair of points problem (<http://www.geek>  
step that takes linear time. Can there be something similar here?

^ | v •



**Vivek VV** • 6 months ago

I had one doubt regarding the method. In step 1/2 how you check for the line s  
the point in the program. Thanks in advance.

^ | v •



**GeeksforGeeks** Mod → Vivek VV • 6 months ago

In BST, we keep active line segments ordered according to their y coo  
we need predecessor in BST. Similarly to find the successor, we need

2 ^ | v .



**Vivek VV** → GeeksforGeeks · 6 months ago

Thanks for the quick reply. I am still not clear :(. When we examine check the intersection with line 2. We only have successor of li not the predecessor of line 3, even by y axis.

1 ^ | v .



**GeeksforGeeks** Mod → Vivek VV · 6 months ago

Vivek, Thanks for pointing this out, intersection of 3 shown updated the example.

^ | v .



**Deepak** → GeeksforGeeks · 3 months ago

Following up on this, which y-coordinate do we use to order line segment has two y-coordinates right. Do we use the

^ | v .



**Deepak** → Deepak · 3 months ago

And if we use the left one, why?

^ | v .



**vivek** · 6 months ago

There is a typo in example dry run.

Following is repeated.

Left end point of line segment 2 is processed:

It should be

Left end point of line segment 3 is processed:

^ | v .



**GeeksforGeeks** Mod vivek · 6 months ago

Thanks for pointing this out. We have updated the example.

^ | v ·



**xxmajia** · 6 months ago

"Building a min heap takes  $O(n)$ "

should be  $O(n \lg n)$ , maintain a heap will cause  $O(\lg n)$ , for building a heap, you  $O(n \lg n)$

1 ^ | v ·



**GeeksforGeeks** Mod xxmajia · 6 months ago

The following wiki link also has proof.

<http://en.wikipedia.org/wiki/B...>

^ | v ·



**GeeksforGeeks** Mod xxmajia · 6 months ago

Building a min heap from given  $n$  elements takes  $O(n)$  time. Please ref

^ | v ·



**xxmajia** GeeksforGeeks · 5 months ago

my bad, it is  $O(n)$ , thanks for correcting me

^ | v ·



Subscribe



Add Disqus to your site



@geeksforgeeks, **Some rights reserved**

**Contact Us!**

Powered by **WordPress** & **MooTools**, customized by geeksforgeeks team