Satik Yeghiazarian

October 25, 2000

CS280A

HW1.4

If we take the shortest chord, then all the points between the endpoints of the shortest chord on the short arc part of the circle belong to distinct chords that intersect the shortest chord. We find the number of chord points on the short arc part of the circle, then delete the shortest chord and continue inductively.

ALGORITHM
Calculate chord lengths for the chords.
Get polar coordinates for all the chords.
    Note: since all the points are on the circle, we only need the angle value for the coordinates. I assume that coordinates are positive, i.e. they go from 0 to 2pi.
Sort chords according to their chord lengths in ascending order and put polar coordinates in array A, i.e. A[1] = (a1, a2), A[2] = (b1, b2) and so on.
For n chords, we have 2*n polar coordinates. Insert the 2*n polar coordinates into Red-Black tree T, i.e. insert a1, a2, b1, b2 and so on.

```
J=1
N=2*n
NumIntChords = 0
While (N>0)   {
    Get polar coordinates of the chord in A[J]          // it's the shortest chord
    Let (a,b) be the polar coordinates.
    NodeX = Search(T, a)
    Rank_A = Rank(T, NodeX)
    NodeY = Search(T, b)
    Rank_B = Rank(T, NodeY)
    // find the number of chord points in the short arc part of the circle, i.e.
    //NumPoints, between points with polar coordinates a and b
    Num1 = ABS(Rank_A - Rank_B) - 1
    Num2 = N - 2 - Num1
    NumPoints = MIN(Num1, Num2)
    //since we took the shortest chord, all the points between a and b belong to
    //distinct chords
    NumIntChords += NumPoints
    //delete the shortest chord
    Delete(T, NodeX)
    Delete(T, NodeY)
    N = N-2
```

```
        //get the next shortest chord from the array A.
        J++
}

return NumIntChords
```

Total time will be O(nlgn) because sorting will take O(nlgn), Search, Rank, Delete for Red-Black tree will take O(lgN), for N elements will take O(NlgN), we can factor the constant out, thus the total time will be in the order of O(nlgn).