4I505 CPA
Welzl's Algorithm

Jordi Bertran de Balanda

February 26, 2016

Abstract

This report presents Welzl's algorithm for obtaining the circle with the smallest radius covering all points in a set. We discuss the merits of this algorithm and compare it to the naive approach to solving the smallest bounding circle problem. We finish by examining the performance of our Java implementation of the algorithm against the naive approach, executing both algorithms on a number of instances of the problem with 256 points in the problem set.

1 Introduction

1.1 Naive approach

The idea behind the naive algorithm to solve the SBC problem is that in instance of the problem with more than 3 points, the correct circle can only be bound by at most 3 points in the problem set. More specifically, the problem set will have either 2 or 3 points on the border of the circle. Therefore, the naive approach first examines whether it is possible to have a circle with only two points on its border that encompasses the whole set. If we find such a circle, it is the optimal circle. If not, we examine every point triplet in the set, and return the triplet which gives the bounding circle with the smallest radius.

1.2 Welzl's Algorithm

The intuition behind Welzl's algorithm is that we can begin with a set containing no points, and, as we add points to the problem set, we can update the smallest bounding circle for the next iteration according to the position of the point in relation to the circle defined in the current iteration. This is expressed by Welzl in a recursive manner that lends itself well to the proof that he provides in his article, but it would be possible to express it in an iterative fashion.

Algorithm 1 Naive Minimum Bounding Circle

```
function NaiveMinCircle(Points)
    $circle = null$
    for p in Points do
        for q in Points do
            $circletmp = Circle(p, q)$
            if $circletmp.rad < circle.radandcircletmp.covers(Points)$ then
                $circle = circletmp$
            end if
        end for
    end for
    if $circle \neq null$ then return $circle$
    end if
    for p in Points do
        for q in Points do
            for r in Points do
                $circletmp = Circle(p, q, r)$
                if $circletmp.rad < circle.rad$ and $circletmp.covers(Points)$ then
                    $circle = circletmp$
                end if
            end for
        end for
    end for
    return $circle$
end function
```

2 Results

2.1 Testing method

We have at our disposal a simple implementation of both the naive algorithm and Welzl's algorithm. To gauge both approaches' performances, we successively run the naive implementation and the Welzl implementation on a series of 1664 instances of 256 point bounding minimum circle problems from the provided Varoumas test base, and we use the JVM's system calls to obtain run time for each algorithm.

Algorithm 2 Naive Minimum Bounding Circle

```
function WelzlRec($P, R$)
    if $P = \emptyset$ or $|R| = 3$ then
        $D$ =boundedCircle($\emptyset$, $R$)
    else
        choose random $p \in Points$
        $D$ =WelzlMinCircle($P - p$, $R$)
        if thenp $\notin D$
            $D$ =WelzlMinCircle($P - p$, $R \cup p$)
        end if
    end if
    return D
end function
function WelzlMinCircle(P)
    return WelzlRec($P$, $\emptyset$)
end function
```

2.2 Performance

3 Analysis

3.1 Expectations

4 Conclusion