# Experimental Document

# Experiment-One Ball-In-Bon

Team Leader: 夏文浩 1605 U201617115 Github-ID:X-wenhao

Team members：

史昊田 1605 U201617111 Github-ID: Shirhockton

付旭宇 1605 U201617102 Github-ID: feature2

张明冶 1605 U201617120 Github-ID: RookieBrain

王之威 1605 U201617112 Github-ID: jillwong

陈 煜 1605 U201617098 Github-ID:ksballetba

姜城阳 1605 U201617105 Github-ID:ganla

Ⅰ、Experiment Introduce

In a square bounded by [-1, 1], given n balloons(they cannot overlap) with variable radio r and position mu. And some tiny blocks are in the square at given position {d}; Balloons can not overlap with these blocks. Find the optimal value of r and mu which maximize sum r^2.

Ⅱ、Experiment Task

Found a team of 7.

Write a user story.

Break down the user story to tasks.

Be familiar with git.

Think out your algorithm to solve the problem.

Write down your test cases.

Start to code with Python.

Use git to store your coding progress and document.

Ⅲ、Mathematical Base Theory

Every condition below, we can determine 0、1 or 2 ball

***3 points***

***2 points*** and ***1 tangent Condition***

***1 point*** and ***2 tangent Conditions***

***3 tangent Conditions***

We use ***Greedy Algorithm.***

The task give us several prerequisite conditions : n points and 4 lines(4 tangent conditions). As a result, we have totally n+4 prerequisite conditions. Apparently, only if every ball is tangent to any other balls, points and lines, the sum of area is man.

***Greedy Algorithm*** of this task:

At the beginning of the task, we have (n+4) conditions. Based on 3 of them, we can determine 1 ball. So there ***C3 x+4 Choices.*** We calculate the area of ball in every situation and chose the man ball. Add it to the Square.

Here comes the point, as we add 1 specific ball, we may consume several conditions (Depends on which conditions we use to determine the 1 specific ball). However, in the meantime, as we have one more ball, we have one added condition(Next added balls must be tangent to this one).

***See point as a extremely tiny ball(which radius is 0)***

★ If we use ***3 points*** to determine a ball:

Lost：3 old conditions;

The number of conditions we have become ***(n+4)-3+1=n+2***;

Method to calculate and determine the ball( x,y are coordinates, r is radius):

***( x - x 1)2 +( y - y 1)2 = ( r + 0 )2 ;***

***( x - x 2)2 +( y - y 2)2 = ( r + 0 )2 ;***

***( x - x 3)2 +( y - y 3)2 = ( r + 0 )2 ;***

Based on the above three equations, we can get a determined ball.

★ If we use 2 ***points and 1 tangent Condition*** to determine a ball:

Lost：2 old conditions(The line is still here);

The number of conditions we have become ***(n+4)-2+1=n+3***;

***( x - x 1)2 +( y - y 1)2 = ( r + r 1)2 ;***

***( x - x 2)2 +( y - y 2)2 = ( r + 0 )2 ;***

***( x - x 3)2 +( y - y 3)2 = ( r + 0 )2 ;***

Based on the above three equations, we can get a determined ball.

★ If we use ***1 points and 2 tangent Condition*** to determine a ball:

Lost：1 old conditions(The line is still here);

The number of conditions we have become ***(n+4)-1+1=n+4***;

***( x - x 1)2 +( y - y 1)2 = ( r + r 1)2 ;***

***( x - x 2)2 +( y - y 2)2 = ( r + r 2)2 ;***

***( x - x 3)2 +( y - y 3)2 = ( r + 0 )2 ;***

Based on the above three equations, we can get a determined ball.

★ If we use ***3 tangent Condition*** to determine a ball:

Lost：No(The line is still here);

The number of conditions we have become ***(n+4)+1=n+5***;

***( x - x 1)2 +( y - y 1)2 = ( r + r 1)2 ;***

***( x - x 2)2 +( y - y 2)2 = ( r + r 2)2 ;***

***( x - x 3)2 +( y - y 3)2 = ( r + r 3)2 ;***

Based on the above three equations, we can get a determined ball.

Every Next step:

Based on the conditions we used in last step, number of conditions we have has changed. Based on the updated conditions, we use the same method to determine next ball.

***In every step, we just add one ball and it is definitely that this decision is best. In the meantime, our next decision is based on last optimal decision. As a result, the final result is optimal.***

Ⅳ、Complexity Analysis

Every Step, we just have to calculate constant times. If we are asked to add N balls, we just have to do ***O( (n+m)3 )*** times compute. Our ***Greedy Algorithm’s complexity is (n+m)3*** . It’s extremely fast and efficient.