**Assignment 3 Report**

**Name:** Akash Shivaji Varude

**Roll Number:** 231110006

1. Implementation Details:

* **Fitness function:**

A screenshot of a computer

Description automatically generatedI calculated Ulysis score for finding fitness of activity\_matrix.

To calculate |Li|, I just traversed activity matrix columnwise and count of columns exactly similar to component ci. Using this, I calculated Wi for each component and stored it in the list W.

To get W\_ulysis, I just calculated average of all of the elements of list W.

* **Suspiciousness function**

I used ochiai score for calculating suspiciousness of a component

Cf: Number of failing tests that execute component

Cp: Number of passing tests that execute component

Nf: Number of failing tests that do not execute component

Initially I set Cf, Cp, Nf all to zero

We already have component vector and error vector also.

Now I traversed through all the values of component and error vector

|  |  |  |
| --- | --- | --- |
| Component value at index i | Value of error vector at index i | Operation |
| 1 | 1 | Cf+=1 |
| 1 | 0 | Cp+=1 |
| 0 | 1 | Nf+=1 |

A screenshot of a computer

Description automatically generatedafter calculating all these values, I calculated final ochiai score using the formula

* **GetRankList Function:**

Let we have [component, suspiciousness\_score] in descending order

e.g.

[['c1', 0.70], ['c2', 0.70], ['c3', 0.66], ['c10', 0.66], ['c5', 0.57], ['c6', 0.57], ['c9', 0.57], ['c4', 0.5], ['c8', 0.5], ['c7', 0.0], ['c11', 0.0]]

Now i have to make ranklist as

[['c1', 2], ['c2', 2], ['c3', 4], ['c10', 4], ['c5', 7], ['c6', 7], ['c9', 7], ['c4', 9], ['c8', 9], ['c7', 11], ['c11', 11]]

**Explanation**:

Here c1 and c2 have same score hence assigned them rank 2.

c3 and c10 have same score hence assign them rank 2+2 i.e. 4

why 4? answer-> two ranks already used. Now there are two components with same score hence assign them rank 4

c5, c6, c9 will have rank=4+3=7 i.e. four ranks already used and there are three components with equal score hence 4+3=7

similarly ranks for other components are also calculated.

**Pseudocode I used** :

1. For each component of activity matrix
   1. Calculate suspiciousness score by calling suspiciousness function
   2. Store the component and it’s score in list as [component, score] in ‘component\_score’ list
2. Sort the component\_score’ in descending order of score (named it ‘Sorted\_Components’)
3. Calculate how many components have same score and store it in ‘same\_score\_id\_count’ list

Explanation:

In above given example, ‘same\_score\_id\_count’ list will look like

[['c1', 2], ['c2', 2], ['c3', 2], ['c10', 2], ['c5', 3], ['c6', 3], ['c9', 3], ['c4', 2], ['c8', 2], ['c7', 2], ['c11', 2]]

This is because component c1 and c2 i.e. only two components have same score 0.70. Hence ['c1', 2], ['c2', 2]

Component c3 and c10 i.e. only two components have same score 0.66. Hence ['c3', 2], ['c10', 2]

Component c5, c6 and c9 i.e. only three components have same score 0.5. Hence ['c5', 3], ['c6', 3], ['c9', 3] and so on.

1. rankList[0]= same\_score\_id\_count[0]
2. Travers i from 1 to m //m is number of components
   1. if(sorted\_components[i][1]!=sorted\_components[i-1][1]):

rankList[i] = [same\_score\_id\_counts[i][0],rankList[i-1][1]+same\_score\_id\_counts[i][1]]

* 1. else:

rankList[i]= [same\_score\_id\_counts[i][0],rankList[i-1][1]]

**Assumption and Limitaiton:**

Evolutionary Search Based Test-suite Generation algorithm given in the assignment can work if there is only 1 error in buggy program in one path

IR of a turtle program is consider as components.

if len(IR) = 5 then components are 0, 1,  2, 3, 4 i.e c0, c1, c2, c3 and c4

**Sample Output:**

A screenshot of a computer

Description automatically generated