Assignment #3: Spectrum Based Fault Localization

Name: Yashvir Singh Nathawat

Roll No: 231110059

Mtech CSE

Email: ysnathawat23@iitk.ac.in

CS639: Program Analysis Verification and Testing

IMPLEMENTATION:

I have implemented this complex task using following basic steps:

1) **Fitness Score**
   Implemented Fitness Function in 2 ways:

   a) **DDU (Density Diversity and Uniqueness) Score**

      - Density of a test suite is summation of all 1's in activation matrix divided by matrix size (Σ Aij / m*n).
      - Diversity ensures no redundant test cases :

$$\delta = 1 - \frac{\sum_{k=1}^{l} |n_k|(|n_k|-1)}{N(N-1)}$$

      - Uniqueness is total number of ambiguity group / Total number of component.
      - Fitness Score = density_bar * Diversity*Uniqueness
        where density_bar = 1 - |1 – 2*density|

b) **Ulysis Score :**

- Formally we can define ulysis score as follows :

$$\mathcal{L}_i = \begin{cases} c_j \mid c_j \in C, j \neq i, & \text{if } c_i = \vec{0} \\ c_j \mid c_j \in C, c_j = c_i, j \neq i, & \text{otherwise.} \end{cases}$$

$$\mathcal{W}_i = \frac{|\mathcal{L}_i|}{m-1}$$

$$\mathcal{W}_{Ulysis} = \frac{1}{m} \sum_{i=1}^{m} \mathcal{W}_i$$

- I have made a dictionary **activity_dict** which contains index with corresponding column entries.
- After that we calculate Wastage Effort for each component and store them in **w_score_list.**
- Divide all members of w_score_list by (num_col – 1).
- Summing all the values from w_score_list and divide by number of columns to get the **fitness_score.**

2) **Suspiciousness:**

- Here we find the suspiciousness score for a given component represented by **comp_index.**
- Here we find 4 metric which are
  - **a)** Cf Number of failing tests that execute C
  - **b)** Cp Number of passing tests that execute C
  - **c)** Nf Number of failing tests that do not execute C
  - **d)** Np Number of passing tests that do not execute C
- After finding above 4 metric, I have used 3 metric to represent suspiciousness of a component :

a) **Ochiai Score:**

Ochiai Score = (cf / (math.sqrt((cf+nf)*(cf+cp))))

b) **Tarantula Score:**

Tarantula Score = (cf / (cf+nf))/ ((cf /(cf+nf)) + (cp / (cp+np)))

c) **Jaccard Score:**

Jaccard _score = cf / (cf + cp + nf)

3) **GetRankList**

- In this function we have find ranks of each components according to their suspiciousness score.
- After calling suspiciousness routine to get the suspiciousness score for each component, we arrange them in a list
- Sorted on the basis of suspiciousness_score and then assigning the rank to each component if two component have same score they have same rank.
- Return the modified ranklist which contains the component name (e.g , ‘c1’) and its rank(an integer)

Assumptions:

The provided code has several assumptions:

1. A float value is anticipated for the fitness score.
2. It is assumed that the one component does not depend on another component.
3. It is assumed that if one component is faulty then its execution will surely result in faulty execution.
4. It is assumed that if no faulty component is executed then program executed correctly.

Limitations:

The provided code has several limitations:

1. The program gives the ranked component list based on the suspiciousness score, there is no guarantee that the component with good rank is guaranteed to be a buggy component.
2. The function determines the fitness score using particular formulas that might not be suitable for all usage scenarios.
3. Fitness Function uses Ulysis Score which is not best for all scenarios but it is good for average case.
4. Suspiciousness Score uses Ochiai metrics, Tarantula and Jaccard metric, all of them are good for certain scenarios and bad for others.