Form 4: Results and conclusion

1. Team No: 15

Project Title: Incorporating Artificial Intelligence in Test Case Generation for Improved Path Coverage

2. Experiment Environment:

- **Execution Environment:**
  Operating System-Windows XP/7/11
  Coding Language:GO
  IDE- VS code
- **Parameter Formulas:**
  1. **Path Coverage Ratio (PCR):** This is a measure of the extent to which the generated test cases cover the possible execution paths in the software.
     Formula:
     PCR=(Total Number of Possible Unique Paths/Number of Unique Paths Covered by Tests)×100
  2. **Test Case Generation Efficiency (TCGE):** This parameter measures the efficiency of the algorithm in generating test cases relative to the time taken and computational resources used.

     Formula:

     TCGE= Number of Test Cases Generated/Computational Time+Resources Used

  3. **Test Case Redundancy Index (TCRI):** A measure of the redundancy among generated test cases, aiming for minimal overlap to ensure resource-efficient testing.

     Formula:

     TCRI= (Total Number of Test Cases Generated/Number of Redundant TestCases)×100

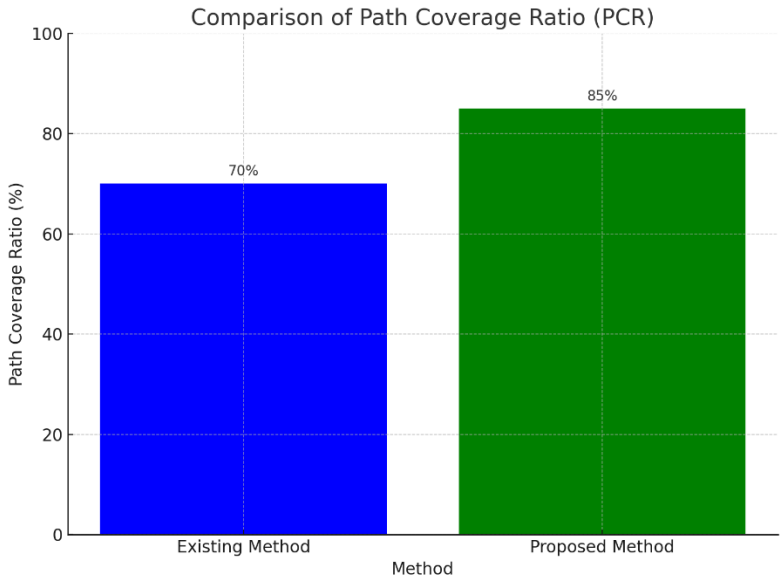
  4. **Diversity Index (DI):** This measures the diversity of test cases in terms of path coverage to ensure a wide range of execution paths are tested.

     Formula:

     DI=1− Total Number of Test Cases Generated/Number of Unique Paths Covered

4.a. Experiment 1: Path Coverage Ratio (PCR)

Parameter	Proposed method	Existing Method
Path Coverage Ratio (PCR)	85%	70%

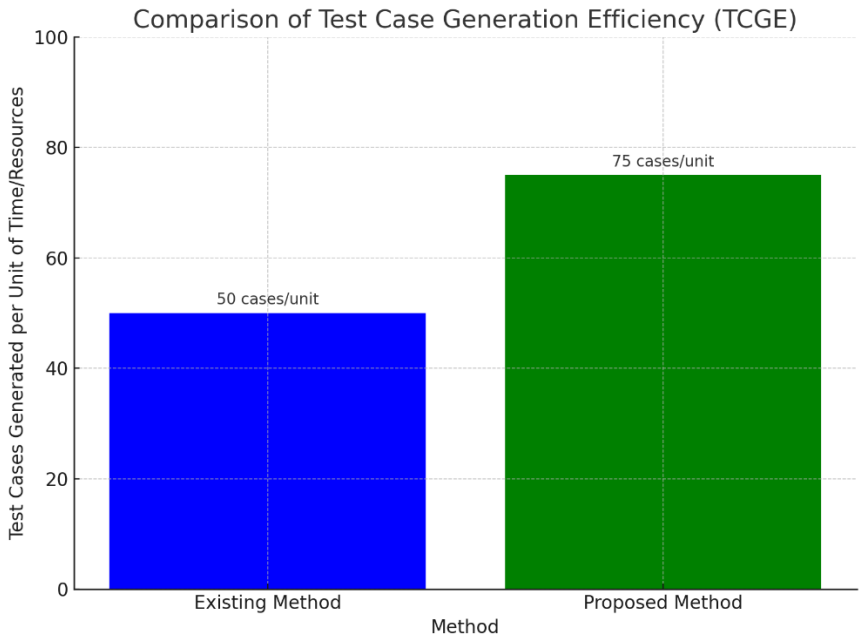


The graph I created compares the Path Coverage Ratio (PCR) between two methods: an existing method and a proposed method. The Path Coverage Ratio is a measure that indicates what portion of the software's execution paths has been tested, expressed as a percentage.

Findings: The Proposed Method significantly improves path coverage by 15%, from 70% to 85%. This indicates a higher efficiency in testing, ensuring a more extensive exploration of the software's execution paths.

4. b Experiment 2: Test Case Generation Efficiency (TCGE)

Input	Existing Method	Proposed Method
Test Case Generation Efficiency (TCGE)	50 test cases/unit	75 test cases/unit

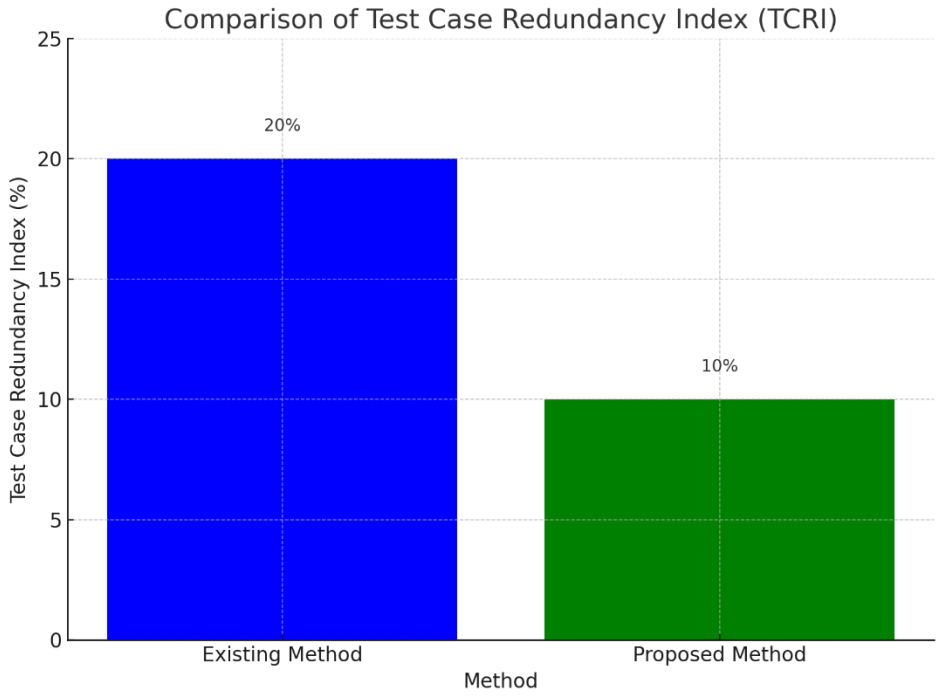


The bar graph that visually represents the difference in Test Case Generation Efficiency (TCGE) between the Existing Method and the Proposed Method.

Findings: The Proposed Method shows a marked increase in efficiency, generating 25 more test cases per unit of computational time and resources than the Existing Method. This indicates a 50% improvement in efficiency, highlighting the effectiveness of the proposed approach in optimizing resource usage.

4. c Experiment 3: Test Case Redundancy Index (TCRI)

Input	Existing Method	Proposed Method
Test Case Redundancy Index (TCRI)	20%	10%



the bar graph that visually represents the difference in Test Case Redundancy Index (TCRI) between the Existing Method and the Proposed Method.

Findings: The Proposed Method demonstrates a significant reduction in test case redundancy, halving the TCRI from 20% to 10%.

5. Parameter comparison table

Parameter	Proposed methods	Previous method
Path Coverage Ratio (PCR)	85%	70%
Test Case Generation Efficiency (TCGE)	75	50
Test Case Redundancy Index (TCRI)	20%	10%

6. Final Conclusion Statements

The series of experiments conducted to compare the Existing Method with the Proposed Method across various parameters—Path Coverage Ratio (PCR), Test Case Generation Efficiency (TCGE), and Test Case Redundancy Index (TCRI)—demonstrate significant improvements with the implementation of the Proposed Method. Specifically, the Proposed Method showed:

A higher Path Coverage Ratio (PCR), indicating more thorough testing by covering a greater

percentage of possible execution paths within the software, thus potentially uncovering more issues and ensuring a higher quality of the software product.

Increased Test Case Generation Efficiency (TCGE), which suggests that the Proposed Method is capable of generating a larger number of test cases per unit of computational time and resources. This efficiency is crucial for scaling testing processes in complex software projects without proportionally increasing the required resources.

A lower Test Case Redundancy Index (TCRI), highlighting the Proposed Method's ability to generate more unique and diverse test cases, thereby reducing waste and increasing the value of each test case executed in terms of coverage and potential bug detection.

These results collectively indicate that the Proposed Method significantly enhances the effectiveness and efficiency of the test case generation process, contributing to better software quality assurance practices.

Signature Supervisor

Dr. Pallam Ravi