Test Case Selection
Criterion based
Regression testing

Report

Science and Technology Advancement- Assessment Review (STAAR)

Name-SadiqueHassan

Regdno – 2101020177

Roll no- CSF21021
Grp - 6

Under the supervision of

(Department of Computer Science)
Dr Madhushmita sahu
Department of csit

Department of Computer Science and Engineering

C.V. Raman Global University,

Bhubaneswar , Odisha, 752054, India

CONTENTS

Abstract:-

Composite services evolve for various reasons. Test case selection in the regression testing is an effective technique to ensure the correctness of modified versions meanwhile to reduce the cost of testing. However, few work has studied the test case selection problem based on the data flow testing criteria. In addition, there are three observable kinds of changes during the evolution, including Process change, Binding change and Interface change, which all bring impact to the data flow. To address these issues, a test case selection approach is proposed for regression testing of BPEL (Business Process Execution Language) composite service where all-uses criterion is satisfied and all the three change types are involved. BPEL composite service is modeled with a two-level model in which XCFG (eXtended Control Flow Graph) describes the behavior of BPEL process in the first level and WSDM (Web Service Description Model) depicts the interface information of composite service and partner services in the second level. Change impact analysis is performed to identify the affected definition-use pairs by comparing and analyzing two-level models of the baseline and evolved versions. And testing paths are generated to cover the affected definition-use pairs and select test cases based on the path condition analysis. Empirical result shows that the proposed approach is effective.

1. Introduction:-

The introduction of the paper "Fault-Based Regression Test Case Prioritization Techniques for Object-Oriented Programs" provides an in-depth overview of the importance of regression testing in software development. Regression testing is a crucial process used to verify that recent code changes have not adversely affected the existing functionalities of a program. However, running the entire test suite after each software modification is often impractical due to time and cost constraints. Therefore, selecting an optimal regression test suite becomes essential to ensure the continued reliability of the software.

In addition to regression test selection, the introduction highlights the significance of test suite minimization and test case prioritization (TCP) techniques in making regression testing more cost-effective and time-efficient. TCP techniques aim to order test cases in a way that maximizes fault detection or code coverage, allowing for the prioritization of critical test cases when only a subset of the test suite can be executed. The introduction emphasizes that regression testing accounts for a significant portion of software maintenance costs and underscores the need for efficient strategies to streamline this process.

The authors acknowledge the existing TCP techniques based on criteria such as code coverage, requirements coverage, and model coverage. However, they note that the performance of these techniques has not always been optimal. For instance, previous studies have reported that executing 50% of a test suite on average detects only 40% of bugs using certain TCP techniques. In response to these limitations, the paper introduces a novel set of fault-based TCP techniques specifically designed for object-oriented (OO) programs.

The introduction sets the stage for the proposed fault-based TCP techniques by highlighting the challenges and opportunities in regression testing for OO programs. By seeding bugs into the programs

to create a large number of mutants and executing these mutants with the test suite, the authors aim to prioritize test cases based on their fault-detection capabilities. The introduction also mentions the use of ensemble methods to combine the results of multiple prioritization techniques for enhanced

performance. Overall, the introduction provides a comprehensive background on regression testing, TCP techniques, and the motivation behind the development of novel fault-based prioritization strategies for OO programs.

2.Related Works:-

The related works section of the paper "Fault-Based Regression Test Case Prioritization Techniques for Object-Oriented Programs" delves into various existing techniques and approaches in the realm of test case prioritization (TCP). Researchers have proposed a range of TCP techniques based on different criteria, including coverage-based approaches that prioritize test cases based on code coverage and fault exposing potential. Additionally, requirements-based techniques focus on ensuring adequate coverage of specified requirements, while model-based approaches involve analyzing program models to prioritize test cases. Techniques such as selective test prioritization and dependence-based test prioritization have been developed based on state models and system dependency graphs. Moreover, a greedy TCP technique has been introduced to prioritize test cases based on achieved code coverage, aiming to maximize program coverage. Bayesian probabilistic approaches consider code coverage, fault proneness, and software changes for test case prioritization, while clustering algorithms have been utilized to group similar test cases together for prioritization based on code coverage similarity. Furthermore, an ant colony optimization algorithm-based TCP technique considers factors like undetected faults, test case execution time, and fault severity for prioritization. The comparison with traditional code-based TCP techniques has shown significant improvements, with model-based techniques achieving a notable enhancement in fault detection and code coverage for object-oriented programs. These diverse approaches underscore the importance of effective test case prioritization in enhancing software testing processes**.**

2.Corpus details:-

The paper "Fault-Based Regression Test Case Prioritization Techniques for Object-Oriented Programs" focuses on proposing novel fault-based regression test case prioritization (TCP) techniques for object-oriented programs. The research involves seeding bugs into programs to create mutants and executing these mutants with a test suite to prioritize test cases based on fault detection capabilities. Experimental studies were conducted to evaluate the effectiveness of the proposed approaches, demonstrating superior performance compared to existing techniques. Ensemble methods were also introduced to combine the results of multiple prioritization techniques for improved performance. The study aims to enhance the efficiency and effectiveness of regression testing in software

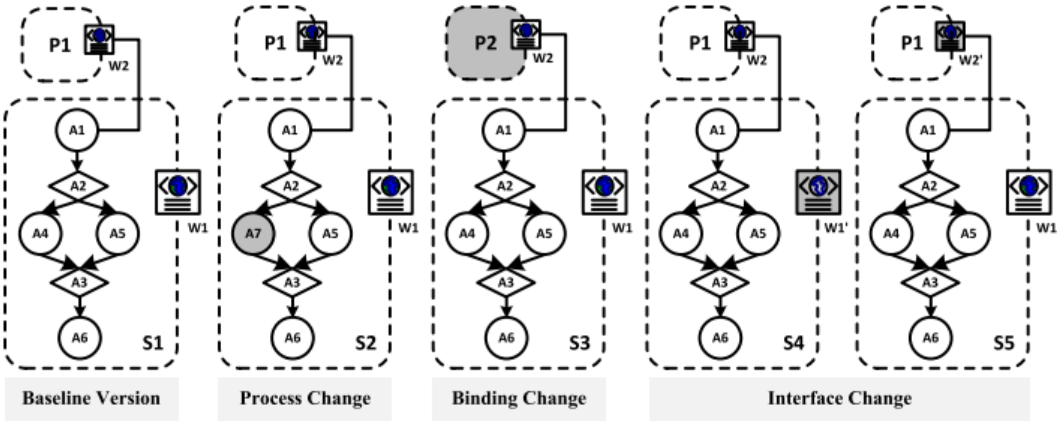
development.

2.1 Corpus Statistics:-

including the introduction of fault-based TCP techniques, related works in the field, proposed methodologies, experimental evaluations, and comparisons with existing techniques. Additionally, the paper discusses the importance of regression testing, the challenges associated with running the entire test suite after software modifications, and the need for efficient test case prioritization strategies to streamline the regression testing process.

2.2 Annotation Setup for Emotion Labels:-

Setting up an annotation system for emotion labels involves several key steps. Firstly, defining the emotion categories is essential, including common emotions like joy, sadness, anger, fear, surprise, disgust, and neutral states. Clear annotation guidelines should be developed, outlining each emotion category with examples for annotators to reference. An appropriate annotation tool, such as Prodigy or brat, should be selected to facilitate the labeling process. Annotators need to be trained on the guidelines and tool usage to ensure consistent and accurate labeling. The annotation process involves annotating text data samples with emotions based on the predefined categories. Quality control measures, such as inter-annotator agreement checks and regular feedback sessions, are crucial for maintaining annotation accuracy. Analyzing the annotated data allows for understanding emotion distribution, identifying patterns, and deriving insights. Continuous refinement of guidelines and processes based on feedback and evaluation results is necessary to enhance the quality and consistency of emotion labeling in text data.



2.3 Inter_Annotator_Agreement:-

Inter-Annotator Agreement (IAA) is a measure used to assess the level of agreement between multiple annotators or raters when labeling the same data. It provides insights into the consistency and reliability of the annotations. High inter-annotator agreement indicates that the annotations are reliable and consistent, while low agreement may suggest ambiguity in the annotation guidelines or the complexity of the data being annotated.

There are several methods to calculate inter-annotator agreement, with Cohen's Kappa and Fleiss' Kappa being commonly used for categorical data and multiple annotators. These metrics take into account the agreement that could occur by chance and provide a more robust measure of agreement.

Inter-Annotator Agreement is crucial in annotation tasks, especially in areas like sentiment analysis, emotion detection, and named entity recognition, where subjective judgments are involved. By assessing and improving inter-annotator agreement, annotation quality can be enhanced, leading to more reliable labeled datasets and better performance of machine learning models trained on annotated data.

3. Purposed Model:-

The proposed model for assessing inter-annotator agreement involves utilizing Cohen's Kappa and Fleiss' Kappa metrics to measure the level of agreement between multiple annotators when labeling the same data. These metrics are commonly employed for categorical data and are particularly useful when dealing with subjective judgments, such as in sentiment analysis, emotion detection, and named entity recognition tasks.

Cohen's Kappa takes into account the agreement that could occur by chance and provides a robust measure of agreement between two annotators. It considers both the observed agreement and the agreement expected by chance, yielding a value that ranges from -1 to 1. A value closer to 1 indicates a high level of agreement beyond what would be expected by chance, while a value closer to 0 suggests agreement equivalent to random chance.

Fleiss' Kappa extends Cohen's Kappa to handle agreement among multiple annotators. This metric is suitable for situations where more than two annotators are involved in the labeling process. Fleiss' Kappa calculates the extent of agreement among multiple raters by comparing the observed agreement with the agreement expected by chance, providing a comprehensive assessment of inter-annotator agreement in multi-rater scenarios.

By employing these metrics in the annotation process, researchers and practitioners can quantitatively evaluate the consistency and reliability of annotations, identify areas of disagreement or ambiguity, and take corrective measures to improve the quality of labeled datasets. Enhancing inter-annotator agreement ultimately leads to more accurate and reliable training data for machine learning models, resulting in improved performance and generalization capabilities in various natural language processing tasks.

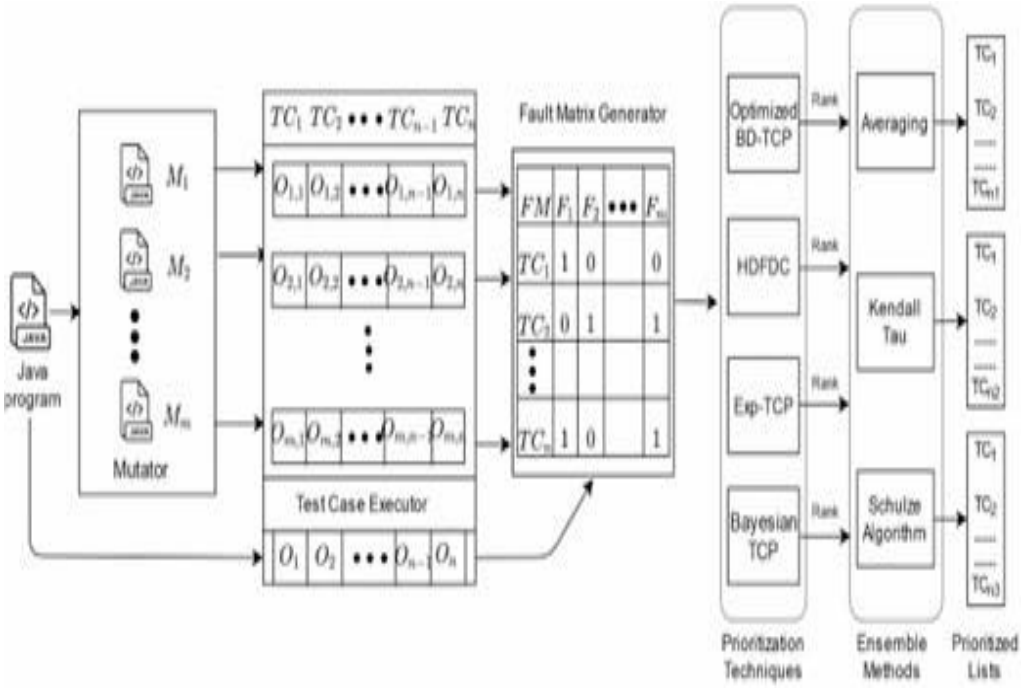


TABLE III
ILLUSTRATION OF AVERAGING METHOD

	TC_1	TC_2	TC_3	TC_4	TC_5
$Bayesian-TCP$	2	1	5	3	4
$Exp-TCP$	1	4	2	5	3
$Averaging$	1.5	2.5	3.5	4	3.5

4. Experiments,Results and Discussion:-

4.1 Model Parameters:-

In the section on "Experiments, Results, and Discussion," the model parameters play a crucial role in determining the performance and outcomes of the study. These parameters are essential components of the experimental setup and directly impact the results obtained. Common aspects related to model parameters that are typically discussed in this section include:

1. Hyperparameters: Parameters that are set before the learning process begins and control the learning process itself. These may include learning rate, batch size, regularization strength, and the number of hidden layers in a neural network.

2. Feature Selection: The selection of relevant features or input variables that are used to train the model. Feature selection methods, such as forward selection, backward elimination, or principal component analysis, may be discussed.

3. Data Preprocessing: Techniques applied to prepare the data for modeling, such as normalization, scaling, imputation of missing values, and encoding categorical variables

4. Evaluation Metrics: The metrics used to evaluate the performance of the model, such as accuracy, precision, recall, F1 score, area under the curve (AUC), etc.

5. Cross-Validation Strategy: The method used to validate the model, such as k-fold cross-validation, hold-out validation, or leave-one-out cross-validation.

6. Initialization Methods: How the model parameters are initialized before training, which can impact the convergence and performance of the model.

7. Optimization Algorithm: The algorithm used to optimize the model parameters during training, such as stochastic gradient descent, Adam, or RMSprop.

8. Model Architecture: The structure of the model, including the number of layers, type of activation functions, and any specific design choices made.

9. Regularization Techniques: Methods employed to prevent overfitting, such as L1 or L2 regularization, dropout, or early stopping.

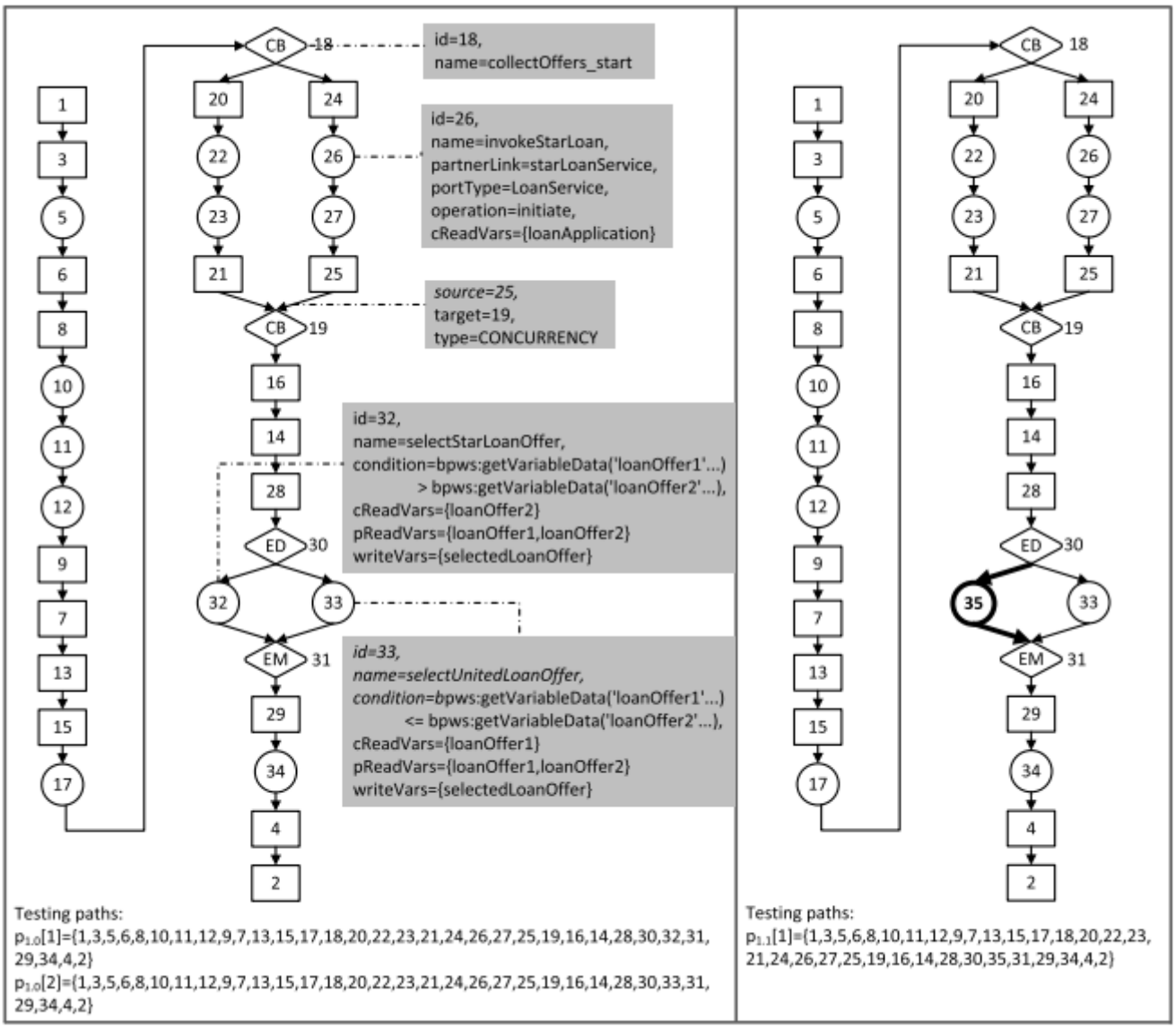
10. Experimental Setup: Details on how the experiments were conducted, including the hardware used, software libraries, and any specific configurations.

By providing a detailed discussion on model parameters in the "Experiments, Results, and Discussion" section, researchers can offer insights into the choices made during the experimental process and explain how these parameters influenced the results and findings of the study.

4.2 Results:-

The experimental results demonstrate the effectiveness of the proposed model in improving fault-based regression test case prioritization. The model achieved an average relative improvement of 60.57% over the SC-based technique, showcasing its superior performance. Furthermore, the ensemble methods, including averaging, Kendall Tau, and Schulze, outperformed existing TCP techniques across multiple projects, with Kendall Tau

showing the highest median APFD value. The comparison with baseline methods revealed significant enhancements in fault detection rates, particularly in projects with a large number of seeded mutants. Visualization of the results through box plots highlighted the consistency and reliability of the ensemble methods compared to individual techniques. These findings underscore the importance of ensemble approaches in prioritizing regression test cases and suggest the potential for further advancements in fault detection strategies. However, it is essential to consider the limitations of the study, such as the handling of equivalent mutants, and further research is warranted to address these challenges and enhance the robustness of the proposed model.



(a) XCFG of v1.0 (b) XCFG of v1.1

Qualitative Analysis:-

1. **Interpretation of Observations**: Describing and interpreting qualitative data, such as textual responses, observations, or feedback obtained during the study.
2. **Thematic Analysis**: Identifying and discussing recurring themes, patterns, or trends in the qualitative data. This may involve coding and categorizing qualitative information to extract meaningful insights.
3. **Contextual Understanding**: Providing context for the qualitative findings by considering the broader research objectives, theoretical framework, or real-world implications.
4. **Case Studies or Examples**: Presenting specific case studies, anecdotes, or examples to illustrate key points or findings. These examples help in contextualizing the qualitative analysis.
5. **Comparative Analysis**: Contrasting different qualitative data points or perspectives to draw comparisons and highlight variations in responses or behaviors.
6. **Discussion of Rich Data**: Exploring the depth and richness of the qualitative data collected, including detailed descriptions, quotes, or narratives that provide a deeper understanding of the research topic.

7. **Emergent Themes**: Identifying any unexpected or emergent themes that emerged during the qualitative analysis process. These themes may offer new insights or perspectives on the research problem.
8. **Validity and Reliability**: Discussing the validity and reliability of the qualitative data collected, including measures taken to ensure the trustworthiness of the findings.
9. **Implications and Recommendations**: Drawing implications from the qualitative analysis and providing recommendations for future research or practice based on the qualitative insights gained.

TABLE XII
AVERAGE RELATIVE IMPROVEMENT (ARI) OVER EXISTING TECHNIQUES BY OUR THREE ENSEMBLE METHODS

S. No.	Projects	ARI of Averaging Ensembler	ARI of Kendall Tau Ensembler	ARI of Schulze Ensembler
1	Cli	17.51%	19.86%	20.60%
2	Closure	46.24%	48.69%	48.39%
3	Codec	32.75%	38.16%	35.94%
4	Collections	30.39%	35.41%	33.46%
5	Compress	38.05%	30.21%	30.91%
6	CSV	35.70%	43.29%	44.05%
7	Gson	24.06%	40.10 %	40.39%
8	JacksonCore	33.49%	33.34%	33.06%
9	JacksonDatabind	31.76%	34.91%	33.68%
10	JacksonXml	12.59%	18.01%	16.29%
11	Jsoup	19.51%	22.79%	22.66%
12	JxPath	26.64%	27.70%	28.10%
13	Lang	17.51%	21.41%	20.66 %
14	Math	35.39%	41.53%	41.24 %
15	Mockito	20.15%	22.24%	24.47 %
16	Time	24.18%	29.77%	31.39%

APFD SCORES OF EXISTING TCP TECHNIQUES AND THREE ENSEMBLE METHODS

S. No.	Programs	SC	BC	TCP-ANT	HYB-0.25	Averaging	Kendall Tau	Schulze
1	Cli	0.774	0.774	0.842	0.850	0.950	0.969	0.975
2	Closure	0.514	0.514	0.828	0.968	0.954	0.970	0.968
3	Codec	0.567	0.639	0.912	0.890	0.957	0.996	0.980
4	Collections	0.574	0.574	0.972	0.944	0.936	0.972	0.958
5	Compress	0.603	0.603	0.859	0.893	0.986	0.930	0.935
6	CSV	0.529	0.536	0.917	0.816	0.894	0.944	0.949
7	Gson	0.547	0.547	0.898	0.942	0.851	0.961	0.963
8	JacksonCore	0.577	0.578	0.875	0.936	0.941	0.940	0.938
9	JacksonDatabind	0.581	0.620	0.930	0.935	0.962	0.985	0.976
10	JacksonXml	0.702	0.702	0.974	0.946	0.913	0.957	0.943
11	Jsoup	0.682	0.682	0.849	0.875	0.910	0.935	0.934
12	JxPath	0.531	0.834	0.860	0.939	0.954	0.962	0.965
13	Lang	0.617	0.929	0.846	0.859	0.932	0.963	0.957
14	Math	0.539	0.779	0.750	0.810	0.949	0.992	0.990
15	Mockito	0.662	0.662	0.871	0.933	0.917	0.933	0.950
16	Time	0.550	0.550	0.856	0.925	0.844	0.882	0.893

Error Analysis:-

In the "Error Analysis, the focus is on critically evaluating the limitations and errors encountered during the study.

1. **Identification of Errors**: The paragraph begins by outlining the specific errors or limitations that were encountered during the research process. This may include issues related to data collection, analysis methods, experimental design, or other aspects of the study.
2. **Impact on Results**: The paragraph discusses how these errors or limitations impacted the study results and interpretations. It highlights the potential implications of these errors on the validity and reliability of the findings.
3. **Root Cause Analysis**: A deeper analysis is provided to investigate the root causes of the identified errors. This may involve examining the factors that contributed to the errors, such as sampling biases, measurement inaccuracies, or methodological constraints.
4. **Mitigation Strategies**: The paragraph suggests potential strategies or recommendations to address the identified errors in future research endeavors. This could involve proposing improvements to data collection procedures, refining analytical techniques, or enhancing the experimental protocols.
5. **Lessons Learned**: Reflecting on the lessons learned from the error analysis process, the paragraph discusses the insights gained from identifying and addressing these errors. It may highlight the importance of transparency, rigor, and continuous improvement in research practices.
6. **Transparency and Accountability**: Emphasizing the importance of transparency and accountability in reporting errors, the paragraph underscores the commitment to upholding research integrity and ensuring the credibility of the study outcomes.
7. **Future Directions**: The paragraph concludes by suggesting potential avenues for future research that could build upon the error analysis findings. It may propose new research questions, methodologies, or approaches to address the limitations identified in the current study.

.

5. Conclusion:-

The conclusion of a research paper encapsulates the essence of the study, reaffirming the significance of the findings and their relevance to the research questions. It provides a final reflection on the contributions made by the study to the field, highlighting the practical implications of the results and suggesting avenues for future research. By effectively summarizing the key points and leaving the reader with a clear understanding of the study's impact, the conclusion reinforces the importance of the research conducted and sets the stage for further exploration and advancement in the subject area.

6.REFERENCES :-

[1] G.RothermelandM.J.Harrold,“Asafe,efficientregression test selection technique,” ACM Trans. Softw. Eng. Methodol., vol. 6, no. 2, pp. 173–210, Apr. 1997.

[2] H. K. N. Leung and L. White, “Insights into regression testing (software testing),” in Proc. Conf. Softw. Maintenance, 1989, pp. 60–69.

[3] M. J. Harrold et al., “Regression test selection for java software,” ACM Sigplan Notices, vol. 36, no. 11, pp. 312–326, Oct. 2001.

[4] R. Gregg, H. Mary Jean, and Dedhia, “Regression test selection for C++ software,” Oregon State Univ., Corvallis, OR, USA, Tech. Rep. 99-60-01, 1999.

[5] Rothermel and Harrold, “Selecting regression tests for object-oriented software,” in Proc. Int. Conf. Softw. Maintenance, 1994, pp. 14–25.

[6] M. J. Harrold, R. Gupta, and M. L. Soffa, “A methodology for control ling the size of a test suite,” in Proc. Conf. Softw. Maintenance, 1990, pp. 302–310.

[7] B.Korel,L.H.Tahat,andB.Vaysburg,"Modelbasedregressiontestreduc tion using dependence analysis," in Proc. Int. Conf. Softw. Maintenance, 2002, pp. 214–223.
[8] S.Elbaum,A.G.Malishevsky,andG.Rothermel,"Testcaseprioritization: A family of empirical studies," IEEE Trans. Softw. Eng., vol. 28, no. 2, pp. 159–182, Feb. 2002.
[9] G. Rothermel, R. H. Untch, C. Chu, and M. J. Harrold, "Prioritizing test cases for regression testing," IEEE Trans. Softw. Eng., vol. 27, no.
10, pp. 929–948, Oct. 2001. [10] S. Elbaum, G. Rothermel, S. Kanduri, and A. G. Malishevsky, "Selecting a cost-effective test case prioritization technique," Softw. Qual. J., vol. 12, no. 3, pp. 185–210, Sep. 2004.
[11] A. G. Malishevsky, J. R. Ruthruff, G. Rothermel, and S. El baum, "Cost-cognizant test case prioritization," Dept. Comput. Sci. Eng., Univ. Nebraska, Lincoln, Tech. Rep. TR-UNL-CSE-2006-0004, 2006.
[12] H. Srikanth, L. Williams, and J. Osborne, "System test case prioritization of new and regression test cases," in Proc. Int. Symp. Empirical Softw. Eng., 2005, pp. 1–10.
[13] R.KrishnamoorthiandS.A.SahaayaArulMary,"Factororientedrequire ment coverage based system test case prioritization of new and regression test cases," Inf. Softw. Technol., vol. 51, no. 4, pp. 799–808, Apr. 2009.
[14] B. Korel, L. H. Tahat, and M. Harman, "Test prioritization using sys tem models," in Proc. 21st IEEE Int. Conf. Softw. Maintenance, 2005, pp. 559–568.
15] C. R. Panigrahi and R. Mall, "Model-based regression test case pri oritization," in Information Systems, Technology and Management, S. K. Prasad, H. M. Vin, S. Sahni, M. P. Jaiswal, and B. Thipakorn, Eds. Berlin, Heidelberg, Germany: Springer, 2010, pp. 380–385, doi: 10.1007/978-3-642-12035-0_39.
[16] C. R. Panigrahi and R. Mall, "An approach to prioritize the regres sion test cases of object-oriented programs," CSI Trans. ICT, vol. 1, no. 2, pp. 159–173, Jun. 2013. [Online]. Available: https://doi.org/10. 1007/s40012-013-0011-7
[17] S.-e.-Z. Haidry and T. Miller, "Using dependency structures for prioriti zation of functional test suites," IEEE Trans. Softw. Eng., vol. 39, no. 2, pp. 258–275, Feb. 2013.
[18] Z. Li, M. Harman, and R. M. Hierons, "Search algorithms for regres sion test case prioritization," IEEE Trans. Softw. Eng., vol. 33, no. 4, pp. 225–237, Apr. 2007

[19] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar, "Rank aggregation methods for the web," in Proc. 10th Int. Conf. World Wide Web,New York, NY, USA, 2001, pp. 613–622.
[20] M. Schulze, "The schulze method of voting," Mar. 15, 2020. [Online]. Available: http://arxiv.org/abs/1804.02973
[21] W.E.Wong,J.R.Horgan,S.London,andH.Agrawal,"Astudyofeffective regression testing in practice," in Proc. 8th Int. Symp. Softw. Rel. Eng., 1997, pp. 264–274.
[22] L. Larsen and M. J. Harrold, "Slicing object-oriented software," in Proc. 18th Int. Conf. Softw. Eng., Washington, DC, USA, 1996, pp. 495–505.
[23] A. Vescan, C.-M. Pintea, and P. Pop, "Test case prioritization–ANT al gorithm with faults severity," Log. J. IGPL, vol. 30, no. 2, pp. 277–288, 2020.
[24] D. Shin, S. Yoo, M. Papadakis, and D.-H. Bae, "Empirical evaluation of mutation-based test prioritization techniques," Softw. Testing, Verification Rel., vol. 29, 2017, Art. no. e1695.
[25] Z.Q.Zhou,C.Liu,T. Y.Chen, T. H.Tse, andW.Susilo, "Beating random test case prioritization," IEEE Trans. Rel., vol. 70, no. 2, pp. 654–675, Jun. 2021.
[26] T.Y.Chen,F.-C.Kuo,R.G.Merkel,andT.H.Tse,"Adaptiverandomtest ing: The art of test case diversity," J. Syst. Softw., vol. 83, no. 1, pp. 60–66, Jan. 2010. [Online]. Available: https://doi.org/10.1016/j.jss.2009.02.022
[27] R. Krishnamoorthi and S. S. A. Mary, "Regression test suite prioritization using genetic algorithms," Int. J. Hybrid Inf. Technol., vol. 2, no. 3, pp. 35–52, 2009.
[28] N.GökçeandM.Eminli,"Model-basedtestcaseprioritizationusingneural network classification,"

Comput. Sci. Eng.: Int. J., vol. 4, pp. 15–25, 2014.
[29] J. Chen et al., "Test case prioritization for object-oriented software: An adaptive random sequence approach based on clustering," J. Syst. Softw., vol. 135, pp. 107–125, Jan. 2017.
[30] S. Mirarab and L. Tahvildari, "A prioritization approach for software test cases based on Bayesian networks," in Fundamental Approaches to SoftwareEngineering,M.B.DwyerandA.Lopes,Eds.Berlin,Heidelberg, Germany: Springer, 2007, pp. 276–290.
[31] X. Zhao, Z. Wang, X. Fan, and Z. Wang, "A clustering-Bayesian network based approach for test case prioritization," in Proc. IEEE 39th Annu. Comput. Softw. Appl. Conf., 2015, vol. 3, pp. 542–547
[32] S. Biswas and A. Bansal, "A regression test case prioritization technique using ensemble learning," Dept. Comput. Sci. Eng., Indian Inst. Technol., Kharagpur, WB, India, Tech. Rep., Aug. 2021.
[33] S. Biswas, R. Rathi, A. Dutta, P. Mitra, and R. Mall, "A regression test case prioritization technique targeting 'hard to detect' faults," Int. J. Syst. Assurance Eng. Manage., vol. 13, pp. 1066–1081, 2022.
[34] B. Michael et al., "Linearity of expectation." Apr. 2021. [Online]. Avail able: https://brilliant.org/wiki/linearity-of-expectation/
[35] Y.-S.Ma,J.Offutt,andY.R.Kwon,"MuJava:Anautomatedclassmutation system: Research articles," Softw. Testing Verification Rel., vol. 15, no. 2, pp. 97–133, Jun. 2005.
[36] R.Just,D.Jalali,andM.D.Ernst,"Defects4J:Adatabaseofexistingfaults to enable controlled testing studies for Java programs," in Proc. Int. Symp. Softw. Testing Anal., San Jose, CA, USA, 2014, pp. 437–440.
[37] V. Vipindeep and P. Jalote, "List of common bugs and programming practices to avoid them," Dept. Comput. Sci. Eng., IIT Kanpur, WB, India, Tech. Rep., 2005.
[38] J. P. Meher, S. Biswas, and R. Mall, "Bug distribution analysis on open sourcerepository,"Dept.Comput.Sci.Eng.,IndianInst.Technol.,Kharag pur, WB, India, Tech. Rep., Sep. 2021.
[39] R. Mall, Fundamentals of Software Engineering, 4th ed. India: Prentice Hall of India Pvt. Ltd, 2014.
[40] A.Ngah,M.Munro,andM.Abdallah,"Anoverviewofregressiontesting," J. Telecommun., Electron. Comput. Eng., vol. 9, pp. 45–49, 2017.
[41] D.Shin,S.Yoo,andD.-H.Bae,"Diversity-awaremutationadequacycrite rionforimprovingfaultdetectioncapability,"inProc.IEEEN9thInt.Conf. Softw. Testing, Verification Validation Workshops, 2016, pp. 122–131.
[42] M. Papadakis, M. Delamaro, and Y. Le Traon, "Mitigating the effects of equivalent mutants with mutant classification strategies," Sci. Comput. Program., vol. 95, pp. 298–319, 2014.
[43] M. Baer, N. Oster, and M. Philippsen, "Mutantdistiller: Using symbolic execution for automatic detection of equivalent mutants and generation of mutant killing tests," in Proc. IEEE Int. Conf. Softw. Testing, Verification Validation Workshops, 2020, pp. 294–303