CS 569 PROJECT IDEA REPORT

AGORA Revisited: A Replication Study on Innovative Test Oracle Generation for Advancing the Robustness of REST API

DONE BY

PARTH SHAH

SANKARANARAYANAN

HARSHINI REDDY KOUKUNTLA

TARUNI MOVVA

Problem:

Our replication study aims to validate and reproduce the findings presented in the paper titled "AGORA: Automated Generation of Test Oracles for REST APIs." The original paper addresses the challenge of automated test oracle generation for REST APIs, highlighting the disparity between the advanced capabilities of automated input generation tools and the simplicity of their test oracles. Despite these tools' proficiency in generating test inputs, their ability to validate and interpret outputs, particularly for complex and nuanced responses, has remained rudimentary. This gap in testing capabilities is where AGORA steps in, offering a solution that significantly enhances test oracle generation.

AGORA, which stands for Automated Generation of Test Oracles for REST APIs, not only addresses this evolving landscape of test case generation tools but also confronts the risks and implications of relying on limited test oracles. These limited oracles, often focusing on crashes, regressions, and violations of API specifications or design best practices, overlook defects and can compromise the integrity of APIs. Such oversights can lead to cascading effects on dependent systems and applications, underscoring the importance of a more robust approach like AGORA.

In our replication study, we intend to build upon the methodology proposed in AGORA to assess its effectiveness and reliability. This involves a deep dive into AGORA's innovative approach to invariant detection, which is a crucial advancement in REST API testing. By detecting a broad spectrum of invariants, AGORA provides a deeper understanding of the diverse behaviors inherent in modern REST APIs, going beyond just identifying more issues. This approach not only aims to address the limitations in current test oracle generation but also enhances the validation of REST API functionality.

Specifically, our study will include:

Replicating the Experimental Setup: We will recreate the experimental conditions outlined in the original paper. This includes using the AGORA approach, the Beet instrumenter, and the Daikon tool, to replicate the methodology and evaluate its effectiveness in a real-world context.

Evaluating Precision and Failure Detection: Our study will measure the precision of AGORA in generating invariants that result in valid test oracles. We aim to assess the impact of the size of the input dataset on AGORA's precision, and how effectively it detects failures, including domain-specific bugs in operations from APIs such as Amadeus, GitHub, Marvel, OMDb, and YouTube.

Verifying Bug Detection: We plan to replicate the detection of domain-specific bugs in various operations from different APIs, verifying AGORA's effectiveness in identifying these bugs and enhancing API reliability.

Examining Categories of Detected Invariants: In our study, we will also analyze the breakdown of detected invariants across different categories as presented in the original paper, further exploring AGORA's capability to understand and interpret the complex behaviors of REST APIs.

Our replication study primarily focuses on addressing the following research questions:

- How effective is AGORA in generating test oracles, and what is the precision of the detected invariants?
- What is the impact of the size of the input dataset on the precision of AGORA?
- How effective are the generated test oracles in detecting failures, especially domain-specific bugs?

To enhance the reproducibility of our study, we plan to utilize the publicly available replication package mentioned in the original paper. This includes access to the source code, data used in the work, and a pre-configured virtual machine to facilitate the replication process. Our replication study aims to contribute to the validation and understanding of AGORA's capabilities in the automated generation of test oracles for REST APIs.

The Design

In the replication study, our primary goal is to reproduce and validate the results presented in the AGORA paper. Here's an overview of the design of the replication experiment:

1) **Objective:**

The main objective is to assess the effectiveness of the AGORA approach for generating test oracles for REST APIs by replicating the experimental setup and methodology described in the original paper.

2) **Replication Package:**

The study uses the publicly available replication package provided by the authors, including the source code, data, and a pre-configured virtual machine. This ensures transparency and facilitates the reproducibility of the experiment.

3) **Experimental Setup:**

Replicate the AGORA approach on the same set of APIs and operations as in the original study, ensuring consistency.
Utilize the same toolset, including Daikon and Beet, to implement the automated generation of test oracles.

4) **API Selection:**

Select a set of industrial APIs that mirrors the ones used in the original study. Ensure diversity in domains and sizes to validate the generalizability of AGORA's effectiveness.

5) **Test Oracle Generation:**

- Reproduce the process of analyzing API requests and responses to detect likely invariants using AGORA.
- Extend Daikon with the same types of invariants and use Beet to instrument the OpenAPI specifications.

6) **Precision and Failure Detection Evaluation**:

- Follow the original study's methodology for evaluating the precision of AGORA in terms of true and false positives in the detected invariants.
- Assess AGORA's effectiveness in detecting failures and identifying real-world bugs in API operations.

7) **Invariant Categories:**

Analyze the breakdown of invariant categories, as presented in Figure 2 of the original paper, to compare and validate the results.

8) **Mutation Testing:**

Replicate the mutation testing process, introducing errors in API responses to assess the robustness of AGORA's test oracles.

9) **Detection of Domain-Specific Bugs**:

Validate the detection of domain-specific bugs in API operations and compare the results with those reported in the original study.

10) **Thorough Review of Specifications:**

Thoroughly review the specifications files, mitigating the potential threat of errors in specifications that deviate from API documentation.

11) **Input Diversity:**

Maximize input diversity by manually selecting a set of varied test inputs for each parameter based on an analysis of the documentation, following the naive and conservative approach described in the original study.

12) **Generalizability Assessment:**

Evaluate the generalizability of the findings by replicating the approach on a set of popular industrial APIs from different domains and sizes.

13) **Addressing Potential Threats**:

Thoroughly address potential threats to validity, including a critical review of specifications, input diversity, and generalizability.

14) **Comparative Benchmarks:**

Include a comparative analysis with other test oracle generation methods to provide a baseline for understanding AGORA's performance relative to existing solutions.

15) **Standardized Reporting:**

Establish clear documentation and reporting standards for the study, detailing criteria for evaluating invariants and processes for classifying true/false positives.

By closely replicating the AGORA approach and experimental setup, the study aims to validate the effectiveness and robustness of the proposed method in generating test oracles for REST APIs. The design prioritizes transparency, consistency, and a thorough evaluation of key metrics, including comparative benchmarks and standardized reporting, to ensure the reliability of the replication results.

Evaluation Plan:

The success of the replication study will be determined by closely aligning with the objectives and metrics outlined in the original AGORA paper. The evaluation plan encompasses data collection, analysis, and validation methodologies.

1) **Evaluation Objectives:**

- Validate the effectiveness of AGORA in generating test oracles for REST APIs.
- Replicate and confirm the results reported in the original paper.
- Assess precision, recall, and overall fault detection capabilities of AGORA.
- Evaluate the generalizability of AGORA's approach across different APIs and domains.

2) Data Collection:

- Utilize the replication package provided by the original authors, including source code, datasets, and virtual machine configurations.
- Collect data on the same set of APIs and operations as used in the original study.
- Ensure consistency in the experimental setup, including the number of API requests, subsets, and categories of invariants.

3) Metrics and Statistical Analysis:

- Evaluate precision and recall of AGORA's invariants by comparing true and false positives.
- Employ statistical tests to assess the significance of differences in results, using appropriate tests such as t-tests or non-parametric equivalents.
- Analyze the breakdown of invariant categories (Figure 2) to validate consistency with the original study.

4) Testing Plan:

- Replicate the AGORA approach by implementing the same modifications to Daikon and utilizing the Beet instrumenter.
- Apply the automated detection of likely invariants to the same set of API requests and responses.
- Verify the precision of detected invariants through manual classification, following the methodology described in the original paper.
- Create behavioral models of the APIs based on collected data to understand the expected behavior and use them as benchmarks for evaluating AGORA’s effectiveness.

5) Fault Detection and Validation:

Validate the effectiveness of AGORA in detecting faults and bugs in API operations. Compare the detected bugs with those reported in the original study, ensuring consistency and reproducibility.

6) Thorough Review and Documentation:

- Thoroughly review the specifications files to ensure accuracy and mitigate potential errors.

- Document the process of replication, including any deviations or challenges encountered.

7) **Diversity and Generalizability:**

- Evaluate the diversity of input API requests and responses to maximize input diversity, following the naive and conservative approach outlined in the original study.
- Assess the generalizability of AGORA's findings by applying the approach to a set of popular industrial APIs from different domains and sizes.

8) **Addressing Potential Threats:**

- Critically assess and address potential threats to validity, including but not limited to errors in specifications, input diversity, and generalizability.
- Document any variations in the experimental setup and discuss their potential impact on the results.

9) **Reproducibility and Transparency:**

- Ensure the entire replication process is documented comprehensively to facilitate reproducibility.
- Utilize a version control system to manage changes to the test setup or methodology and provide a clear history of the study's evolution.

10) **API Response Time Measurement:**

In addition to precision and recall, measure the response time of APIs when tested with AGORA to understand its impact on API performance.

The success of the evaluation will be determined by the ability to reproduce the results presented in the original paper, validating the effectiveness of the automated generation of test oracles for REST APIs through the detection of likely invariants. The use of consistent metrics, statistical analyses, behavioral models, and a thorough testing plan, including response time measurement and version control, will contribute to a robust and reliable evaluation of AGORA's capabilities.

<u>Project Timeline:</u>

We, as a team, have outlined a planned timeline for our replication study

1. Initial Understanding (January 24 - February 2):

- Review the original paper thoroughly to understand its objectives, methodology, and findings.
- Identify the key components of the study, including the research questions, experimental setup, and evaluation metrics.

2. Data Preparation (February 3 - February 8):

- Collect the necessary data, including the API specifications, requests, responses, and any other relevant information.
- Ensure that we have access to the tools and software mentioned in the original paper, such as Daikon and AGORA.

3. Reproduction of Experiment (February 9 - February 25):

- Set up the experimental environment, replicating the tools and configurations used in the original study.
- Execute the experiment as described in the paper, generating test oracles and evaluating the performance of AGORA.

4. Results Analysis (February 26 - March 2):

- Analyze the results obtained from our replication, comparing them with the original study.
- Identify any discrepancies or differences in outcomes and thoroughly document them.

5. Documentation (March 3 - March 7):

- Prepare a comprehensive report outlining the replication process, detailing the methodology, results, and any variations from the original study.

• Include visualizations, tables, or graphs to illustrate our findings.

• Clearly document any challenges or limitations encountered during the replication.

6. Feedback and Final Adjustments (March 8 - March 10):

• Share our replication report with the Professor and peers for an independent review

• Gather feedback on the validity and accuracy of our replication.

• Make any final adjustments to our replication study based on feedback received during the reviews.

7. Submission (March 11):

• Submit our replication study by the specified deadline.

• Include all relevant documents to support our work.

<u>Group Member Responsibilities :</u>

This section outlines the planned responsibilities for each team member, providing a roadmap for the upcoming phases of the project. It offers a proactive glimpse into our organized approach to collaboration.

1. Parth Shah - Team Lead and Data Collection:

Manages overall project operations, coordinates team efforts, sets milestones, ensures deadlines are met, and reports progress. Leads technical development and system integration for AGORA replication.

2. Sankaranarayanan - Research and Development Specialist:
Conducts in-depth research and analysis of AGORA's architecture, comprehends its key components and functionality, implements core algorithms, integrates new data formats, and upholds code quality.

3. Harshini - Testing and Evaluation Lead:

Oversees all testing activities, designs and executes test cases, develops test plans, evaluates AGORA's performance through testing, and analyzes test results.

4. Taruni - Document and Reporting Analyst:

Responsible for all project documentation and reporting, writes technical reports, documents development and testing processes, and prepares presentations for stakeholders.