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

Presentation By Parth Shah, Sankaranarayanan, Harshini Koukuntla, Taruni Movva

January 29, 2024

Introduction

Our replication study seeks to reproduce and validate the findings of a previous research project that introduced an innovative tool named AGORA. AGORA, an acronym for Automated Generation of Test Oracles for REST APIs, is designed to automate the API testing process by generating specialized entities known as "test oracles."

- The key feature of AGORA is its utilization of invariants—properties of the API output that should consistently hold. This is achieved by analyzing previous API requests and their corresponding responses to learn the expected behavior of the API.

Why is AGORA Valuable?

- Efficiency in Testing:

AGORA makes the testing process faster and more efficient. Instead of manually checking if an API is behaving correctly, AGORA does it automatically.

- Learning from Examples:

AGORA learns from previous examples of how an API should behave. It doesn't just rely on predefined rules; it can adapt and understand new patterns.

- Finding Mistakes Early:

AGORA helps catch mistakes in APIs before they cause problems in real-world applications. It acts like a watchful eye, ensuring that the APIs work smoothly.

- Wide Applicability:

AGORA isn't limited to a specific type of API. It can be used with various APIs, making it a versatile and valuable tool for developers across different projects.

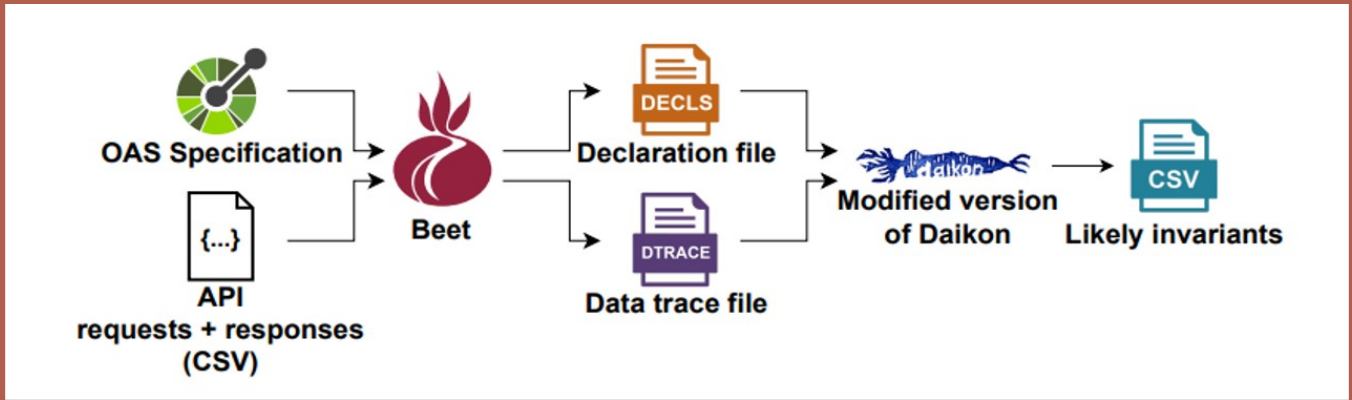
The Current Challenge in REST API Testing

- **Limited Scope of Existing Test Oracles**: Traditional test oracles in REST API testing have been limited in scope, primarily focusing on detecting straightforward issues like crashes, regressions, and API specification violations. This narrow focus leaves a gap in identifying more complex, nuanced failures that are often critical to the API's functionality and user experience.

- **Manual and Time-Consuming Processes**: The creation and configuration of test oracles have conventionally been a manual process, requiring substantial time and effort. This manual intervention not only slows down the testing process but also introduces the risk of human error, impacting the overall effectiveness and efficiency of the testing.

How AGORA Works:

AGORA uses a tool called Daikon to analyze the behavior of APIs. It looks at examples of API requests and responses, figures out the expected results, and creates test oracles based on that understanding.
Workflow of AGORA



How AGORA Benefits Developers:

Developers can use AGORA to ensure their APIs are error-free and meet the desired specifications. It saves time, reduces manual effort, and improves the overall quality of software applications.

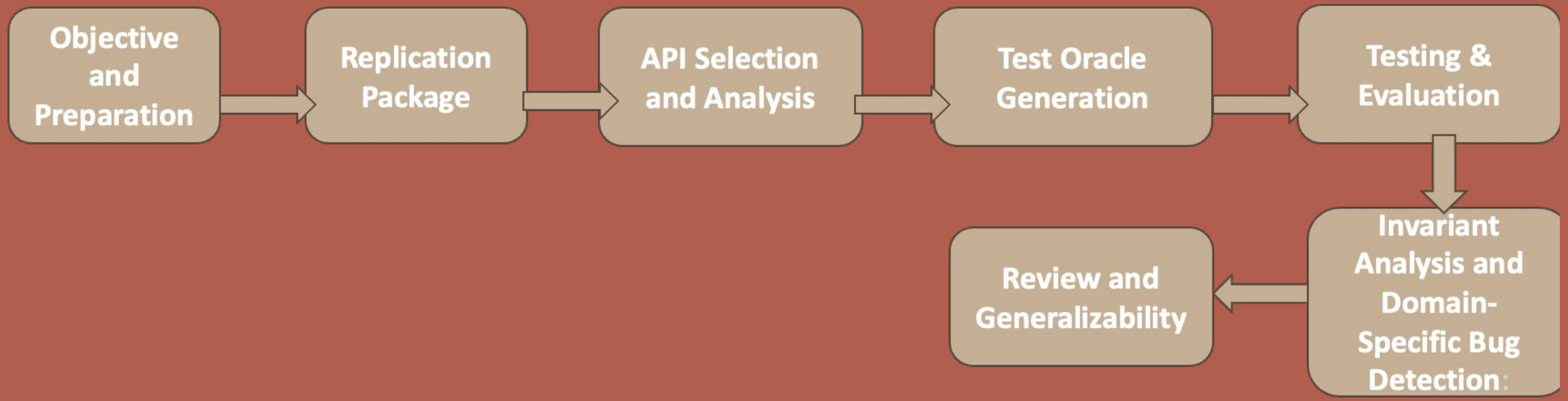
Research Questions

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?

Workflow and Design

The replication study's design aims to validate AGORA's method for creating test oracles for REST APIs. Key steps include using a provided replication package for consistency, replicating AGORA on selected APIs, and evaluating the generated test oracles' precision and failure detection. The study also involves mutation testing, detecting domain-specific bugs, reviewing API specifications, ensuring diverse inputs, assessing the results' generalizability, and addressing any validity threats. This rigorous approach ensures the reliability of the replication outcomes.



Evaluation Plan

Here's a concise evaluation plan for the replication study:

- **Objective Alignment:** Confirm AGORA's test oracle effectiveness for REST APIs and replicate original paper's results.
- **Data Consistency**: Use the provided package for source code and datasets (AmadeusHotel, GitHub, Marvel, and YouTube) focusing on identical APIs and operations.
- **Testing Execution**: Implement AGORA's approach with Daikon and Beet, verify invariant precision manually.
- **Fault Detection**: Validate AGORA's bug detection, compare with original findings.
- **Specification Review**: Ensure accuracy in API specifications, document all replication steps.
- **Input Diversity:** Maximize test input variance and assess AGORA's application to various industrial APIs.
- **Threat Mitigation**: Address validity threats, document any setup variations and their impact.
- **Reproducibility Focus**: Document for transparency, ensuring reproducibility of the study.

Roles and Responsibilities

Name	Role	Responsibilities
Parth Shah	Project Lead	• Overall project management, coordination between team members • Setting milestones, ensuring deadlines are met, and reporting progress. • Leading the technical development of AGORA replication, including coding and system integration.
Sankar Narayan	Research and Development Specialist	• Research and Analyze AGORA's existing architecture, ensuring a thorough grasp of its core components and functionality • Implementing AGORA's core algorithms, integrating new data formats, and maintaining code quality.
Harshini Koukuntla	Testing and Evaluation Lead	• Overseeing the testing process, including the design and execution of test cases. • Developing test plans, conducting tests to evaluate AGORA's performance, and analyzing test results.
Taruni	Document and Reporting Analyst	• Handling all documentation and reporting aspects of the project. • Writing technical reports, documenting the development and testing processes, and preparing presentations for audience.

Timeline

Timeframe	Activity	Team Member
Week 1-3	• Initial research and analysis of AGORA • Requirement gathering	• All Team members
Week 3-6	• Development of feature enhancements • First phase of testing and feedback • Refinement of developed features based on feedback	• Team Lead • Research and development specialist
Week 6-9	• Compilation of testing results and analysis • Second phase of testing and evaluation • Drafting of final report and presentation	• Testing and Evaluation Lead • Documentation and Reporting analyst
Week 10	• Review and finalization of report and presentation • Final presentation and dissemination of findings	• All Team members

Thank you