УДК 04

Аксёнова Д.А.

Студент 2 курса магистратуры

Поволжский государственный университет телекоммуникаций и информатики

Россия, г. Самара

ИССЛЕДОВАНИЕ АСПЕКТОВ ТЕСТИРОВАНИЯ ПРОГРАММНОГО КОДА С ИСПОЛЬЗОВАНИЕМ SMT-РЕШАТЕЛЕЙ

Аннотация: В статье рассматриваются различные аспекты тестирования программного кода с использованием SMT-решателей (Satisfiability Modulo Theories). Использование SMT-решателей для поиска ошибок в программном коде и доказательства его корректности по отношению к заданным спецификациям и свойствам. Эта публикация будет полезна разработчикам программного обеспечения, которые заинтересованы в повышении надежности и качества своего кода, а также специалистам по тестированию, которые ищут методы оптимизации процесса тестирования.

Ключевые слова: SMT-решатель, тестирование кода, методы тестирования (модульное, интеграционное, системное, приемочное), статический анализ кода, дилемма статического анализа.

Aksenova D.A.

Second year graduate student

Povolzhskiy State University of Telecommunications and Informatics

Russia, Samara

STUDYING ASPECTS OF PROGRAM CODE TESTING USING SMT SOLVERS

Annotation: The article discusses various aspects of testing program code using SMT solvers (Satisfiability Modulo Theories). Using SMT solvers to find errors in program code and prove its correctness in relation to given specifications and properties. This publication will be useful to software developers who are interested in improving the reliability and quality of their code, as well as testing professionals who are looking for methods to optimize the testing process.

Keywords: SMT solver, code testing, testing methods (unit, integration, system, acceptance), static code analysis, static analysis dilemma.

Тестирование кода — это процесс проверки программного кода на соответствие определенным требованиям и ожиданиям. Основная цель тестирования состоит в обнаружении ошибок и дефектов в программном обеспечении до его выпуска, чтобы уменьшить вероятность возникновения проблем в реальной эксплуатации.

В процессе тестирования применяются различные методы, включая модульное тестирование, интеграционное тестирование, системное тестирование, приемочное тестирование и другие. Каждый из этих методов обладает своими особенностями и применяется на различных этапах разработки программного обеспечения.

С развитием выявления уязвимостей и дефектов в языках программирования стали появляться статистические анализаторы, способные автоматически обнаруживать подозрительные участки кода по заранее заданным шаблонам. Такие анализаторы легко интегрируются в среду разработки и могут предупреждать о возможных проблемах. Однако существует известная проблема статического анализа - дилемма: либо применяются строгие методы, которые избегают пропуска ошибок, но

могут порождать множество ложных срабатываний, либо уделяется приоритет поиску точного списка проблем, что может привести к упущению реальных ошибок.

Для проверки соответствия логики исходного кода заявленным функциональным требованиям часто формируются специальные команды, которые используют различные инструменты для тестирования кода. Однако иногда бывает сложно гарантировать, что алгоритм всегда достигнет ожидаемого результата во всех возможных случаях, и не всегда можно убедиться, что учтены все возможные состояния программы. Многолетний опыт показывает, что тестирование кода может быть ошибочным наравне с самой программой. Поэтому для проверяющего эксперта было бы желательно наличие инструмента, который позволил бы проводить проверку в автоматическом режиме.

Одним из инструментов, который может быть полезен, это программа, которая преобразует исходный код в скрипт, представляющий программу в формате, совместимом с SMT-решателем. Тестированию будут подвергаться только публичные методы, поскольку только они доступны для вызова пользователями. Чтобы убедиться, что все ожидаемые состояния могут быть достигнуты, а некорректные состояния недостижимы при любых условиях, будет достаточно следующего набора данных:

- Переменные, которые участвуют в алгоритме, включая внутренние переменные метода.
- Логические выражения изменения каждой из этих переменных. Берем во внимание от каких переменных зависит, при каких условиях изменяется и как.
- Условие перехода. Условием перехода является выполнение логических выражений всех переменных в правильной последовательности.

- Начальное состояние системы - набор значений для всех переменных программы после ее инициализации.
- Конечное состояние системы - набор значений для переменных программы, которые должен получить решатель.
- Количество переходов, через которое программа должна пройти, чтобы система из начального состояния превратиться в конечное.

Программа автоматически извлекает или создает все данные, кроме окончательного состояния системы и количества переходов, на основе исходного кода, а остальные данные запрашивает у пользователя. После этого программа генерирует скрипт для SMT-решателя, который затем пытается найти условие, при котором состояние программы после определенного числа переходов будет эквивалентно заданному конечному состоянию.

Использованные источники:

1. Э. Кларк, О. Грамберг, Д. Пелед. Верификация моделей программ: Model Checking. М.: МЦНМО, 2002 .
2. Вельдер С. Э., Лукин М. А., Шалыто А. А., Яминов Б. Р. SAT/SMT by Example. СПбГУ ИТМО, 2011. 3.
3. А. С. Камкин. Введение в формальные методы верификации программ: учебное пособие – Москва: МАКС Пресс, 2018.