Санкт-Петербургский государственный университет

Исследование инструментов фаззинга для генерации модульных тестов на Java

Осипова Александра Вадимовна
группа 17.Б11-мм

Научный руководитель: к.т.н., доцент кафедры СП Ю. В. Литвинов
Консультант: инженер ООО «Техкомпания Хуавей» В. Е. Володин
Рецензент: руководитель отдела ООО «Техкомпания Хуавей» Д. А. Иванов

20 мая 2021

Суть работы

- Модульные тесты повышают качество кода
  - Писать вручную (много шаблонного кода, отнимает ценное время программиста)
  - Генерировать автоматически
- Идея
  - Создание теста (вызов методов и создание аргументов для них)
  - Подбор значений аргументов с помощью фаззинга

Цели и задачи

Цель – создание системы, совмещающей инструменты фаззинга и генерации модульных тестов для Java, и исследование их совместной работы

Задачи:

- создать систему, совмещающую инструменты фаззинга и генерации модульных тестов, выбрать инструменты фаззинга и генерации тестов и интегрировать их в созданную систему
- наладить её работу в инфраструктуре Java Unit Testing Tool Competition
- произвести сравнение совместной работы выбранных инструментов, сопоставить их работу с другими инструментами для генерации модульных тестов

Существующие решения

- Случайное тестирование
  - T3
  - Randoop
    - случайная генерация с обратной связью
- Тестирование на основе поиска
  - Sushi, Tardis
  - EvoSuite
    - эволюционный алгоритм
- Символьное исполнение
  - CATG, SPF, EvoSuiteDSE

Фаззеры

- Метод чёрного ящика
  - JQF + No Guidance
- Метод белого ящика
  - jFuzz
- Метод серого ящика
  - Kelinci
  - java-afl
  - javaFuzz
  - JQF + Zest

Процесс генерации нового теста

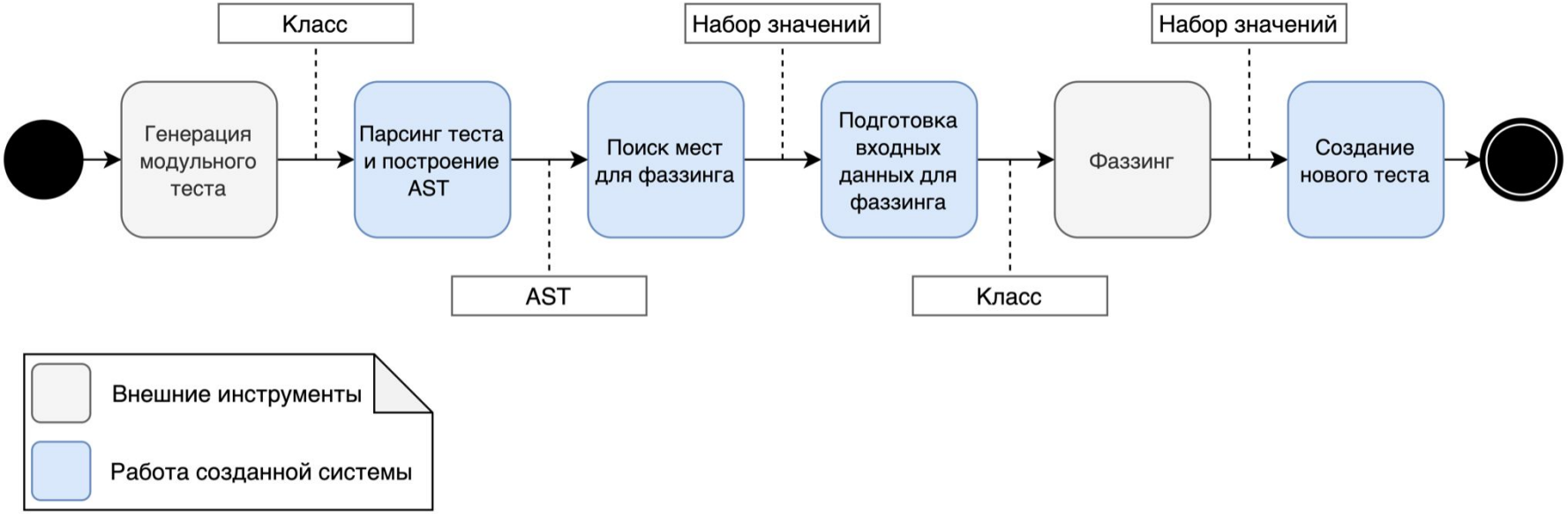

Рис. 1: Диаграмма активностей процесса создания теста с модифицированными значениями

Архитектура системы

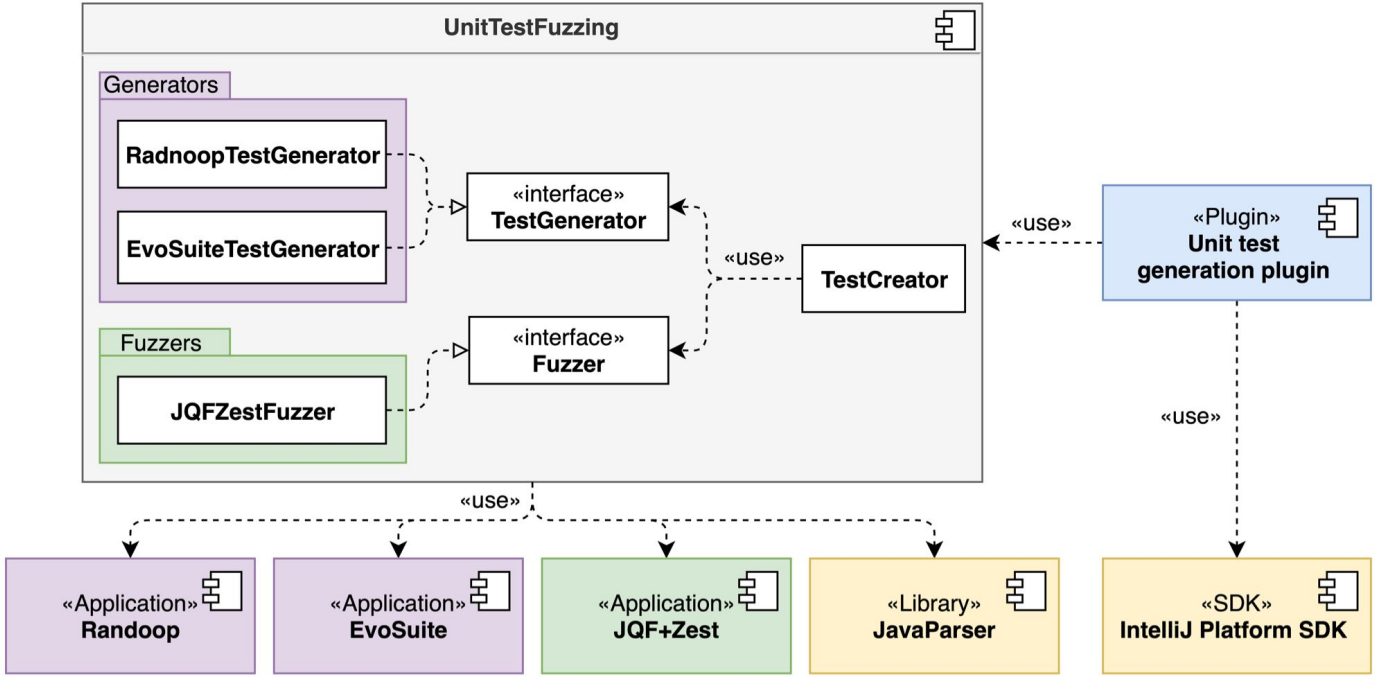

Рис. 2: Диаграмма компонентов и основных классов системы

Java Unit Testing Tool Competition

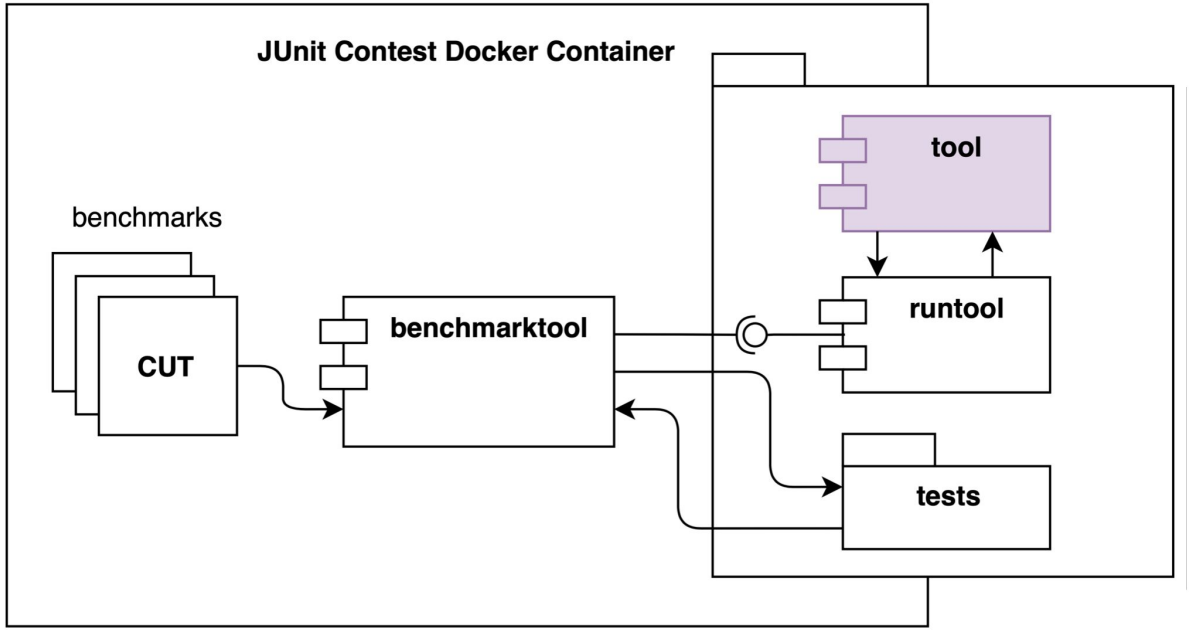

Рис. 3: Диаграмма компонентов архитектуры Java Unit Testing Tool Competition

Постановка экспериментов

- 6-Core Intel Core i7, 2.2 GHz, RAM DDR4 16Gb, macOS Big Sur 11.2.3
- Набор данных
  - Fescar/Seata
  - Guava
  - Spoon
- Временной бюджет
  - 30 секунд
  - 300 секунд
- Метрики
  - Покрытие строк кода
  - Покрытие условий

Результаты экспериментов

Инструмент	Временной бюджет, с	Среднее покрытие строк кода, %	Среднее покрытие условий, %
Randoop	30	40.62	28.70
Randoop/JQF+Zest	300 (30/270)	40.62	28.93
Randoop	300	58.28	50.19
EvoSuite	30	60.38	53.07
EvoSuite/JQF+Zest	300 (30/270)	60.38	53.07
EvoSuite	300	68.19	58.63

Таб. 1: Результаты проведённых экспериментов

Интерпретация результатов

- В контексте соревнований и небольших временных бюджетов применение фаззинга не даёт желаемых результатов
  - Большое количество сгенерированных методов для фаззинга требует колоссальных вычислительных ресурсов для фаззинга каждого метода
  - Выделенного на фаззинг одного метода времени может быть недостаточно, чтобы улучшить покрытие
  - Накладные расходы при совмещении инструментов

Результаты

В рамках данной работы:

- создана система, совмещающая инструменты генерации тестов и фаззинга
  - добавлены генераторы Randoop, EvoSuite
  - добавлен фаззер JQF + Zest
- налажена работа в инфраструктуре Java Unit Testing Tool Competition
- проведены эксперименты

Ссылка на проект: https://github.com/Software-Analysis-Team/Unit-Test-Fuzzer