Ранжирование фрагментов кода по важности

Субботина Олеся Вадимовна
Руководитель: Резник Сергей Александрович

НИУ ВШЭ - Санкт-Петербург

18 июня 2021 г.

Мотивация

- Идея:
  - В больших проектах много кода, есть более важные фрагменты, есть менее важные
  - Хотим отранжировать их по важности, по строгому критерию и автоматически
- Возможные применения:
  - Помощь в массовых исследованиях кодовых разных проектов
  - Сбор статистики о проектах
  - Помощь в планировании работы, учёбы

Цель и задачи

- Цель:
  - Расширяемый gradle плагин с возможностью проведения нескольких вариантов анализа кода. Исходя из результатов анализов, получение пользователем списка ранжированных по важности фрагментов кода заданного им проекта.
- Задачи:
  - Статический анализ
  - Расширяемый gradle плагин
  - Динамический анализ

Аналоги

- Точных аналогов проекта нет.
  Тем не менее, существуют некоторые смежные средства для анализа кода:
  - Измерители покрытия (coverage tools): Jacoco, Jcov
  - Профайлеры, т.е. сборщики характеристик работы программы: JProfiler, YourKit

Статический анализ

- Статический анализ заданного пользователем проекта с использованием библиотеки ASM
- Построение графа зависимостей между методами классов проекта
- Ранжирование вершин графа, алгоритм Pagerank

Плагин

Мотивация:

- Хочется оставить разработчикам возможность расширять проект и работать с их собственным представлением графа зависимостей
- В будущем хочется добиться инвариантности относительно языка программирования

Реализация:

- Реализация загрузчика пользовательских классов для построения графов
- Оформление проекта в виде плагина для gradle

Динамический анализ

- Инструментирование байткода с использованием библиотеки javassist
  - Отслеживание инструкций байткода, связанных с вызовами методов
  - Вставка вызова метода из класса ”сборщика информации” в байткод
- Сбор информации посредством запуска пользователем проекта
- Анализ полученных данных
  - Построение графа зависимостей
  - Ранжирование вершин графа, алгоритм Pagerank

Сравнение результатов

Результаты, полученные при статическом, динамическом и экспертном анализах:

StaticCodeRankingStat.txt

```
Class rank: 0.06953
ru.hse.projects.maze.MazeSolver

Class rank: 0.03797
ru.hse.projects.maze.OutputPath

Class rank: 0.02221
ru.hse.projects.maze.Launcher
ru.hse.projects.maze.IndexPair

Class rank: 0.01730
ru.hse.projects.maze.Main
ru.hse.projects.maze.Vertex
ru.hse.projects.maze.DebugOutput
```

DynamicCodeRankingStat.txt

```
Rank: 0.04747
ru.hse.projects.maze.MazeSolver

Rank: 0.04042
ru.hse.projects.maze.OutputPath

Rank: 0.02926
ru.hse.projects.maze.IndexPair

Rank: 0.02319
ru.hse.projects.maze.Vertex

Rank: 0.02142
ru.hse.projects.maze.Launcher
ru.hse.projects.maze.Main
ru.hse.projects.maze.DebugOutput
```

ExpectedOutput.txt

```
MazeSolver
OutputPath
IndexPair
Launcher
Vertex
Main
DebugOutput
```

Результат

- Расширяемый gradle-плагин для Java
- Возможность проведения статического анализа кода проекта
- Возможность проведения динамического анализа кода проекта
- Предоставление пользователю списка ранжированных по важности классов проекта

Планы на будущее

- Доработка динамического анализа
- Расширение плагина и инвариантность относительно языка
- Добавление метрик ранжирования
- Ускорение и оптимизации

Репозиторий

Ссылка на репозиторий:
github.com/OlesiaSub/CodeRanking