

Проект Architecture Reloaded: поиск дефектов в архитектуре

Тимофей Брыксин

18 сентября 2017 г.

МОТИВАЦИЯ

Проект Machine Learning for Solving Software Maintenance and Evolution Problems.
Университет Бабеш-Бойяи, Румыния

- ▶ *Marian, Z.* Aggregated metrics guided software restructuring. IEEE International Conference on Intelligent Computer Communication and Processing (ICCP), 2012
- ▶ *Marian, Z. Czibula, G., Czibula, I.G.* Using Software Metrics for Automatic Software Design Improvement. Studies in Informatics and Control, 21(3), 2012
- ▶ *Marian, Z.* A study on hierarchical clustering based software restructuring. Studia Universitatis Babes-Bolyai, Informatica 57 (2), 2012

Подход №1: построение метрики для системы в целом

- ▶ Подсчёт 16 метрик
- ▶ Агрегированная метрика класса
 - ▶ выбор оптимальных и предельных значений
 - ▶ приведение всех метрик к $[0, 1]$, чем меньше, тем лучше
 - ▶ сумма всех нормализованных метрик
- ▶ Агрегированная метрика системы
 - ▶ среднее значение агрегированных метрик классов

Задача кластеризации

1. каждый метод помещаем в свой кластер (класс)
2. повторяем, пока не останется один кластер
 - ▶ рассматриваем все варианты слить два кластера; выбираем тот, который даёт меньшую агрегированную метрику системы
 - ▶ принимаем получившееся разбиение за текущее
3. из всех получившихся разбиений на шагах 1 и 2 выбираем разбиение с минимальной агрегированной метрикой

Пример

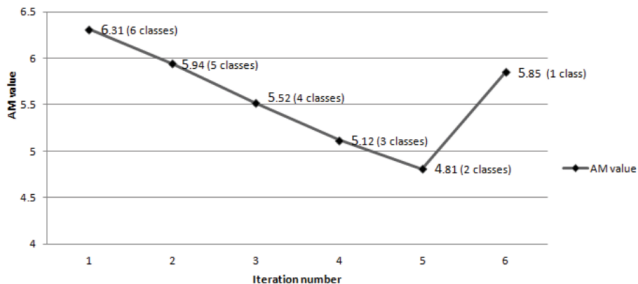
```
1. public class Class_A {
2.     public static int attributeA1;
3.     public static int attributeA2;
4.     public static void mA1(){
5.         attributeA1 = 0;
6.         mA2();
7.     }
8.     public static void mA2(){
9.         attributeA2 = 0;
10.        attributeA1 = 0;
11.    }
12.    public static void mA3(){
13.        attributeA2 = 0;
14.        attributeA1 = 0;
15.        mA1();
16.        mA2();
17.    }
18. }
```

```
20. public class Class_B {
21.     private static int attributeB1;
22.     private static int attributeB2;
23.     public static void mB1(){
24.         Class_A.attributeA1=0;
25.         Class_A.attributeA2=0;
26.         Class_A.mA1();
27.     }
28.     public static void mB2(){
29.         attributeB1=0;
30.         attributeB2=0;
31.     }
32.     public static void mB3(){
33.         attributeB1=0;
34.         mB1();
35.         mB2();
36.     }
37. }
```

Результаты

No.	Partition	Value for metric <i>AM</i>
1	$\{\{mA1()\}, \{mA2()\}, \{mA3()\}, \{mB1()\}, \{mB2()\}, \{mB3()\}\}$	6.31
2	$\{\{mA1()\}, \{mA2(), mA3()\}, \{mB1()\}, \{mB2()\}, \{mB3()\}\}$	5.942
3	$\{\{mA1()\}, \{mA2(), mA3()\}, \{mB1()\}, \{mB2(), mB3()\}\}$	5.52
4	$\{\{mA1(), mB1()\}, \{mA2(), mA3()\}, \{mB2(), mB3()\}\}$	5.12
5	$\{\{mA1(), mB1(), mA2(), mA3()\}, \{mB2(), mB3()\}\}$	4.81
6	$\{\{mA1(), mB1(), mA2(), mA3(), mB2(), mB3()\}\}$	5.85

AM value changes



Подход №2

- ▶ Каждому классу и методу ставится в соответствие вектор свойств
- ▶ Вводится функция расстояния между этими векторами
- ▶ На базе неё осуществляется кластеризация

Компоненты вектора

- ▶ Relevant properties
 - ▶ для метода: сам метод, объемлющий класс, все поля объемлющего класса и методы системы, используемые в методе, все методы системы, перекрывающие метод
 - ▶ для класса: сам класс, все его поля и методы, все реализуемые им интерфейсы, все его классы-наследники
- ▶ Depth in Inheritance
- ▶ Number of Children
- ▶ Fan-In
- ▶ Fan-Out

Полу-метрика

$s_i = (s_{i1}, s_{i2}, s_{i3}, s_{i4}, s_{i5})$ – вектора свойств (RP + метрики)

$$d(s_i, s_j) = \begin{cases} 0, & \text{если } i = j \\ \sqrt{\frac{1}{m} \cdot \left(1 - \frac{|s_{i1} \cup s_{j1}|}{|s_{i1} \cap s_{j1}|} + \sum_{k=2}^m (s_{ik} - s_{jk})^2 \right)}, & \text{если } s_{i1} \cap s_{j1} \neq \emptyset \\ \infty, & \text{иначе} \end{cases}$$

Automatic Refactoring Identification (ARI)

Начинаем с пустого разбиения. Для каждого метода:

1. ищем такой класс C , что $d(m, C) < d(m, C') \forall C' \in S, C \neq C'$
2. если $d(m, C) < 1$, то можно определить рефакторинг Move Method:
 - ▶ если C находится в некотором элементе разбиения k , то добавляем m туда же
 - ▶ если C не входит в разбиение, создаём новый кластер и добавляем в него C и m
3. если $d(m, C') = \infty \forall C' \in S$, то это рефакторинг Extract Class. Ищем метод m' , ближайший к m
 - ▶ если такой нашёлся, помещаем m в кластер к нему
 - ▶ если $d(m, m') = \infty \forall m' \in S$, создаём новый кластер (класс) для m
4. сливаем все классы C_1, C_2 , что $d(C_1, C_2) < 1$ (рефакторинг Inline Class)

Эксперименты

Пример с Move Method

	Class_A	Class_B	mA1()	mA2()	mA3()	mB1()	mB2()	mB3()
Class_A	0	∞	0.329	0.397	0.528	0.471	∞	∞
Class_B	∞	0	∞	∞	∞	0.482	0.555	0.239
mA1()	0.329	∞	0	0.361	0.453	0.406	∞	∞
mA2()	0.397	∞	0.361	0	0.596	0.464	∞	∞
mA3()	0.528	∞	0.453	0.596	0	0.444	∞	∞
mB1()	0.471	0.482	0.406	0.464	0.444	0	0.474	0.471
mB2()	5.49	0.555	∞	∞	∞	0.474	0	0.567
mB3()	5.5	0.239	∞	∞	∞	0.471	0.567	0

$$K_1 = \{Class_A, mA1(), mA2(), mA3(), mB1()\}$$

$$K_2 = \{Class_B, mB2(), mB3()\}$$

Hierarchical Agglomerative Clustering (HAC)

Иерархическая агломеративная кластеризация с методом полной связи и эвристической функцией слияния кластеров

- ▶ $D(K_1, K_2) = \max_{x \in K_1, y \in K_2} d(x, y)$
- ▶ кластера сливаются только если расстояние между ними меньше 1

Большой пример

Фреймворк JHotDraw: 173 класса, 1375 методов, 475 полей классов

- ▶ ARI: найдено 20 рефакторингов Move Method, 9 некорректных
- ▶ HAC: найдено 158 кластеров, из них 3 новых
 - ▶ всего 12 методов, 5 некорректно перемещённых

Программная реализация

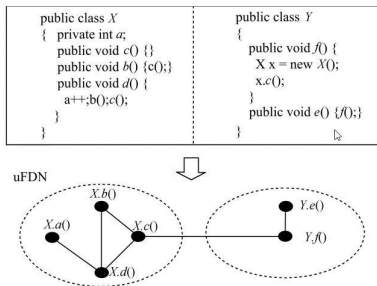
- ▶ Авторская реализация FAOS (закрытая и недоступная ☹)
- ▶ Плагин MetricsReloaded для IDEA
 - ▶ Наборы метрик Martin Packaging, MOOD, Chidamber-Kemerer, LOC, Complexity и др.
 - ▶ <https://github.com/BasLeijdekkers/MetricsReloaded>

Наши эксперименты на JHotDraw 5.1

- ▶ ARI: 91 рефакторинг против 20
- ▶ HAC: 85 рефакторингов против 12

CCDA: Constrained Community Detection Algorithm

- ▶ *Wei-Feng Pan, Bo Jiang, Bing Li. Refactoring Software Packages via Community Detection in Complex Software Networks // International Journal of Automation and Computing, 2013, pp.157–166.*



- ▶ Максимизация показателя качества $Q = \sum_{i=1}^n (e_{ii} - a_i^2)$

FMC: Fields and Methods Clustering

- ▶ Илия Кузьмина, дипломная работа, 2017 г.
- ▶ Совмещение иерархической кластеризации и метода K-средних
 - ▶ случайная инициализация центров кластеров
 - ▶ вычисление расстояний от каждой сущности до всех кластеров (метод полной связи)
 - ▶ перемещение сущности с ближайшим расстоянием в соответствующий кластер
 - ▶ пока не останется нераспределённых сущностей
 - ▶ повтор итерации с разбиением N раз

Эксперименты и модификации

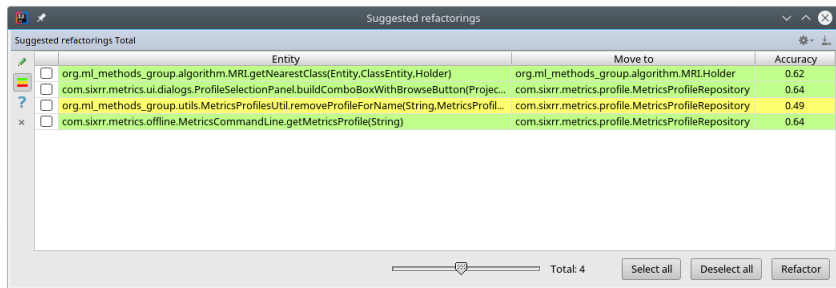
- ▶ Состав множеств Relevant Properties
- ▶ Введение весов для свойств Relevant Properties
- ▶ Разные ОО-метрики
- ▶ Пересечения результатов алгоритмов
- ▶ Механизм голосования

Сопутствующие инструменты

- ▶ Gradle, CI, testing
- ▶ Распараллеливание
- ▶ UI для отображения результатов
- ▶ Инструменты для применения рефакторингов
- ▶ Инструменты для отладки и оценки результатов

ArchitectureReloaded

- ▶ <https://github.com/ml-in-programming/ArchitectureReloaded>



Идеи на будущее

- ▶ Аннотатор
- ▶ Применение подхода Label propagation
- ▶ Ещё раз попробовать поиграться с весами
- ▶ Другие алгоритмы кластеризации
- ▶ Другие фишки полей и методов