

Detecting software defects using relational rules mining

Симиютин Борис

научный руководитель: Тимофей Брыксин

СПб АУ НОЦНТ РАН

1 июня 2017 г.

Detecting software design defects using relational
association rule mining
<http://bit.ly/2rlv5DN>

Как нам формально определить, хороший код, или нет?

Как нам формально определить, хороший код, или нет?

- Берем некоторый набор метрик

Как нам формально определить, хороший код, или нет?

- Берем некоторый набор метрик
- Определяем сущности нашего кода, например, классы

Как нам формально определить, хороший код, или нет?

- Берем некоторый набор метрик
- Определяем сущности нашего кода, например, классы
- Переводим их в вектора из пространства метрик

Как нам формально определить, хороший код, или нет?

- Берем некоторый набор метрик
- Определяем сущности нашего кода, например, классы
- Переводим их в вектора из пространства метрик
- Что-то делаем

Association rules

- Пример: "Если покупатель купит хлопья для завтрака, то скорее всего он купит и молоко."
- В нашем случае: "Если между метриками установилось какое-то соотношение, то скорее всего оно характеризует код"
- Например, $a_1 \leq a_2 \leq a_3$, где a_1, a_2, a_3 - какие-то метрики

Формализм

- Association rule:
 $(a_1, a_2, a_m) \Rightarrow (a_1\mu_1 a_2\mu_2 \dots a_{m-1}\mu_{m-1} a_m)$, где a_i - атрибуты записи, $\{\mu_i\}$ - отношения. Например, $\{<, >, =\}$
- Support: $Support(X)$ = процент сущностей, в которых есть все атрибуты из правила
- Confidence: $Confidence(X \Rightarrow Y)$ = процент сущностей из $Support(X)$, на которых выполняется правило
- Правило $X \Rightarrow Y$ называется *интересным*, если $Support(X) \geq s_{min}, Confidence(X \Rightarrow Y) \geq c_{min}$

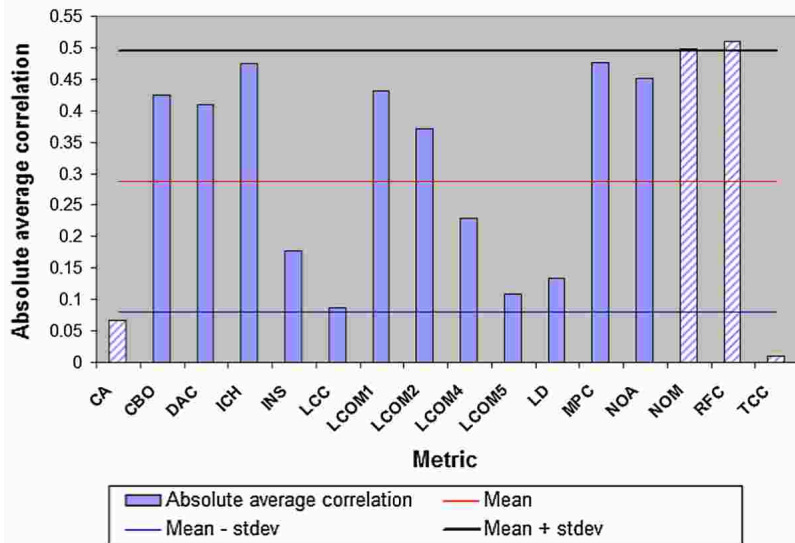
Алгоритм

- 1 Среди набора метрик отбираем перспективные
- 2 Ищем интересные правила
- 3 Определяем критерии, по которым будем отделять хороший код от плохого

Алгоритм обучения: отбор метрик

- 1 Pearson correlation coefficient
- 2 $avg(sm_i)$ - среднее значение корреляции между sm_i и всеми остальными метриками
- 3 $mean = mean(avg), stddev = stddev(avg)$
- 4 Оставляем только те, для кого $|avg(sm_i) - m| > stddev$

Алгоритм обучения: отбор метрик. пример



- ① Выбираем множество отношений, например, $\{<, >, =\}$
 - ① Ищем все интересные правила длины 2
 - ② Ищем все интересные правила длины 3
 - ③ Оставляем наиболее длинные правила
 - ④ Ищем все интересные правила длины 4
 - ⑤ ...
 - ⑥ Пока на новой итерации не нашли новых правил

Алгоритм построения: поиск правил. пример

```
public class Class_A {  
    public static int attributeA1;  
    public static int attributeA2;  
  
    public static void mA1(){  
        attributeA1 = 0;  
        mA2();  
    }  
  
    public static void mA2(){  
        attributeA2 = 0;  
        attributeA1 = 0;  
    }  
  
    public static void mA3(){  
        attributeA2 = 0;  
        attributeA1 = 0;  
        mA1();  
        mA2();  
    }  
}
```

```
public class Class_B {  
    private static int attributeB1;  
    private static int attributeB2;  
  
    public static void mB1(){  
        Class_A.attributeA1=0;  
        Class_A.attributeA2=0;  
        Class_A.mA1();  
    }  
  
    public static void mB2(){  
        attributeB1=0;  
        attributeB2=0;  
    }  
  
    public static void mB3(){  
        attributeB1=0;  
        mB1();  
        m2();  
    }  
}
```

	Length	Rule	Confidence
Найденные правила:	2	$DIT > NOC$	1
	2	$NOC < FI$	0.625
	2	$FI > FO$	0.5
	3	$DIT > NOC < FI$	0.625
	3	$DIT > NOC < FO$	0.75
	3	$NOC < FI > FO$	0.5
	4	$DIT > NOC < FI > FO$	0.5
	Length	Rule	Confidence
После сокращения:	3	$DIT > NOC < FO$	0.75
	4	$DIT > NOC < FI > FO$	0.5

Метрики:

- Depth of inheritance tree(DIT)
- Number of children(NOC)
- Fan-In(FI)
- Fan-Out(FO)

- 1 Пусть на вход дали множество сущностей S_{new}
- 2 Проверяем на них все правила
- 3 Находим множество P потенциально ошибочных сущностей - тех, для кого процент ошибок p_e выше порога
- 4 Отбираем из них тех, у кого p_e выше среднего по множеству
- 5 Если $P = \emptyset$, то отбираем тех, у кого $n_e(s_i) > meanErr(S_{new}) + stddevErr(S_{new})$

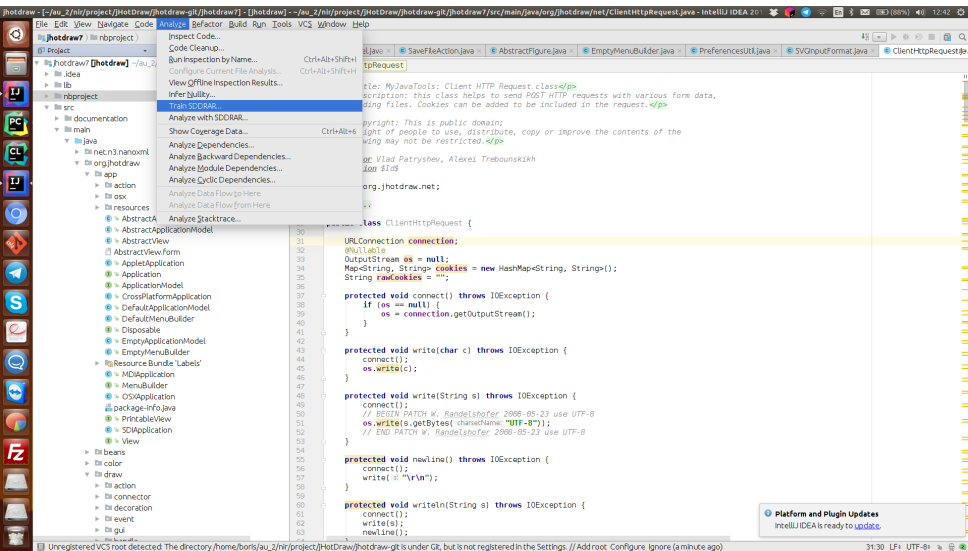
Реализация

- Алгоритм реализован на Java

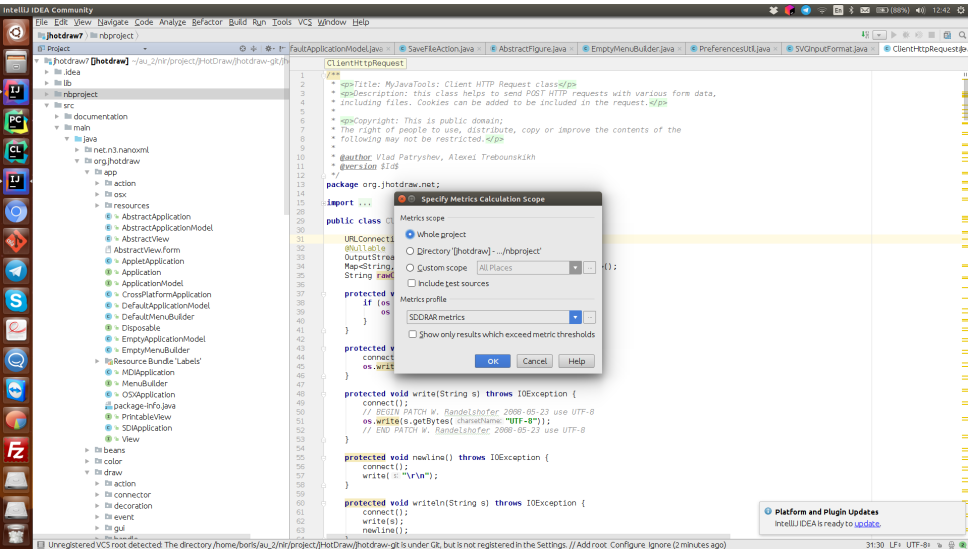
- Алгоритм реализован на Java
- В качестве кодового окружения использован плагин MetricsReloaded для IDEA

- Алгоритм реализован на Java
- В качестве кодового окружения использован плагин MetricsReloaded для IDEA
- С новыми метриками, добавленными в другом НИРе

Использование



Использование



Использование

IntelliJ IDEA 2017.2.4 (96%) 12:57

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

Project: hotdraw7 [hotdraw7] - [au_2/nlr/project/hotdraw/hotdraw-gtk/hotdraw7] - [hotdraw7] - [au_2/nlr/project/hotdraw/hotdraw-gtk/hotdraw7/src/main/java/org/hotdraw/net/ClientHttpRequest.java - IntelliJ IDEA 2017.2.4 (96%) 12:57

ClientHttpRequest.java

```
50 // BEGIN PATCH W. Randselshofer 2008-05-23 use UTF-8
51 os.write(s.getBytes( charsetName "UTF-8"));
52 // END PATCH W. Randselshofer 2008-05-23 use UTF-8
53 }
54
55 protected void newline() throws IOException {
56     connect();
57     write("\r\n");
58 }
59
60 protected void writeLn(String s) throws IOException {
61     connect();
62     write(s);
63     newline();
64 }
65
66 private static Random random = new Random();
67
68 protected static String randomString() { return Long.toString(random.nextLong(), radix 36); }
69 String boundary = "-----" + randomString() + randomString() + randomString();
70
71 private void boundary() throws IOException {
72     write(boundary);
73 }
74
```

Metrics SDRAR metrics for Project: hotdraw7 from Cp, 31 Jan 2017 12:55:01 MSK

Class metrics

class	CBO	Command	Cons	CSA	CSQ	CSQA	DAG	Dcy	DIT	E	ICH	LCOM	LCOM1	LCOMS	Level	Level*	NAAC	NOM	OCavg	OCmax
net.n3.nanoml.CDATAReader	4	2	1	5	31	36	1	2	2	53 488	0	1	0	0.17	2	2	3	4	4.25	9
net.n3.nanoml.ContentReader	6	2	1	6	31	37	2	5	2	104 231	0	1	0	0.12	5	5	4	4	4.00	8
net.n3.nanoml.LDMLBuilder	1	1	1	1	1	1	0	1	1	1	0	1	1	1.00	1	1	1	8	1.00	1
net.n3.nanoml.LDMLElement	1	1	1	1	1	1	0	1	1	1	0	1	1	1.00	1	1	1	51	1.00	1
net.n3.nanoml.LDMLEntityResolver	1	1	1	1	1	1	0	1	1	1	0	1	1	1.00	1	1	1	4	1.00	1
net.n3.nanoml.LDMLParser	1	1	1	1	1	1	0	1	1	1	0	1	1	1.00	1	1	1	9	1.00	1
net.n3.nanoml.LDMLReader	1	1	1	1	1	1	0	1	1	1	0	1	1	1.00	1	1	1	13	1.00	1
net.n3.nanoml.LDMLValidator	1	1	1	1	1	1	0	1	1	1	0	1	1	1.00	1	1	1	8	1.00	1
net.n3.nanoml.NonValidator	7	13	1	3	35	38	1	6	1	1 399 354	10	3	43	0.64	5	5	3	15	4.20	15
net.n3.nanoml.LPReader	3	2	1	4	31	35	1	2	2	22 037	0	1	0	0.25	2	2	2	4	3.25	7
net.n3.nanoml.Scd0MLBuilder	5	8	2	3	31	34	1	4	1	127 683	1	3	5	0.48	4	4	3	11	2.00	5
net.n3.nanoml.Scd0MLParser	11	13	1	4	40	44	4	11	1	2 355 809	12	1	0	0.61	6	6	4	19	3.53	25
net.n3.nanoml.Scd0MLReader	6	6	3	2	46	48	1	2	1	964 930	11	2	0	0.47	2	2	2	21	2.48	11
net.n3.nanoml.Scd0MLReader.StackedRee	1	0	0	4	12	16	0	0	1	0	0	0	0	0.59	1	1	4	0	1.00	1
net.n3.nanoml.ValidatorPlugin	5	17	1	1	40	41	1	5	1	44 112	0	9	58	0.00	5	5	1	20	1.00	1
net.n3.nanoml.XMLAttribute	1	1	1	5	19	24	0	0	1	664	0	5	7	0.70						
net.n3.nanoml.XMLElement	8	12	5	11	124	135	1	3	1	2 327 695	43	7	1 272	0.85						
net.n3.nanoml.XMLEntityResolver	5	3	1	1	23	24	0	3	1	24 486	3	1	0	0.00						

Platform and Plugin Updates
IntelliJ IDEA is ready to update.

62:18 LF: UTF-8

Результаты

Интересные метрики:

NumOperationsAdded
NumTransitiveDependencies
DepthOfInheritance
CouplingBetweenObjects
NumAttributesAdded
LackOfCohesionOfMethods
NumCommands
MaximumOperationComplexity
LevelOrder
AverageOperationParameters

NumOperationsOverridden
NumInterfacesImplemented
ClassSizeAttributes
NumDependencies
NumDependents
ClassSizeOperations
AdjustedLevelOrder
HalsteadEffort
ClassSizeOperationsAttributes
NumOperationsInherited

Результаты

FTP4j		ISO8583	WinRun4j
MySDDRAR	FTPClient(God)	MessageFactory(God)	Closure(one_big_method)
	FTPFile(God) Base64		FileVerb Launcher NativeBinder(God) NativeHelper Native(interesting_case_only_native_methods) PInvoke,PInvoke.NativeStruct Registry.FILETIME,Registry.QUERY_INFO(public_not_final_data_fields) RegistryKey
SDDRAR	FTPClient	MessageFactory	NativeBinder
iPlasma	FTPClient FTPFile	MessageFactory	Closure FileVerb NativeBinder
JDeodorant	FTPClient NVTASCIIWriter NVTASCIIReader FTPProxyConnector FTPCommunicationChannel	MessageFactory ISOMessage	Launcher FileAssociation RegistryKey

Что еще хотелось:

- Анализировать, почему помеченные классы оказались помеченными, и классифицировать по типичным архитектурным промахам
- Оформить в виде отдельного плагина