

论文阅读报告-crx

论文: **Structured information in bug report descriptions—influence on IR-based bug localization and developers**

作者: 程荣鑫

日期: 2021.3.27

本论文作者研究了bug reports中的结构化信息 (src codes & stack traces) 对IR模型的影响。

作者研究的数据集中, 有7334篇bug报告, 其中30%包含结构化信息, 这与我们的swtBugReport类似, 他们的思路是有可能在我们的项目中复用的。

我在文章中提取了一些自认为有助于软工三IRBL项目的内容, 下面我来一个个解释。

1. Bug report tagging

1. 依据一或多个空行, 对bug报告进行分段
2. 使用正则表达式匹配堆栈信息段, 将该段落标记为ST
3. 使用正则表达式: class定义正则、method定义正则、if/while/for、赋值语句正则, 匹配Java code, 将源代码段标记为SC
4. 剩下的语段标记为NL

2. BLUiR和AmaLgam

这两个也是IR模型, BLUiR的思路是将**注释和变量名方法名语句**抽离开来, 分别计算二者和bug报告summary&description的准确度, 这个思路是可以借鉴的, 因为我们一阶段是将注释和代码混在一起计算与bug报告相似度的。

AmaLgam则是对BLUiR的改进, 它在后者基础上考虑当前bug报告和以往修复bug报告之间的相似度。在该模型中, 认为:

1. 以往的相似的报告中的修复文件
2. 最近修改的文件 (版本历史)

上述二者可能引入bug

其中第2点从目前来看我们没有数据支持, 所以可以尝试吸纳第一条思路, 计算报告与报告之间的相似度, 并在推测目标源文件时考虑该相似度。

3. ST or SC

在作者的实验中，NL+ST+SC（也就是完整的使用bug报告），这样的定位效果是最差的，我们需要删除掉bug report中的一些冗余成分。

NL是不能删除的，根据论文中的观点，这是最有价值的一类信息。

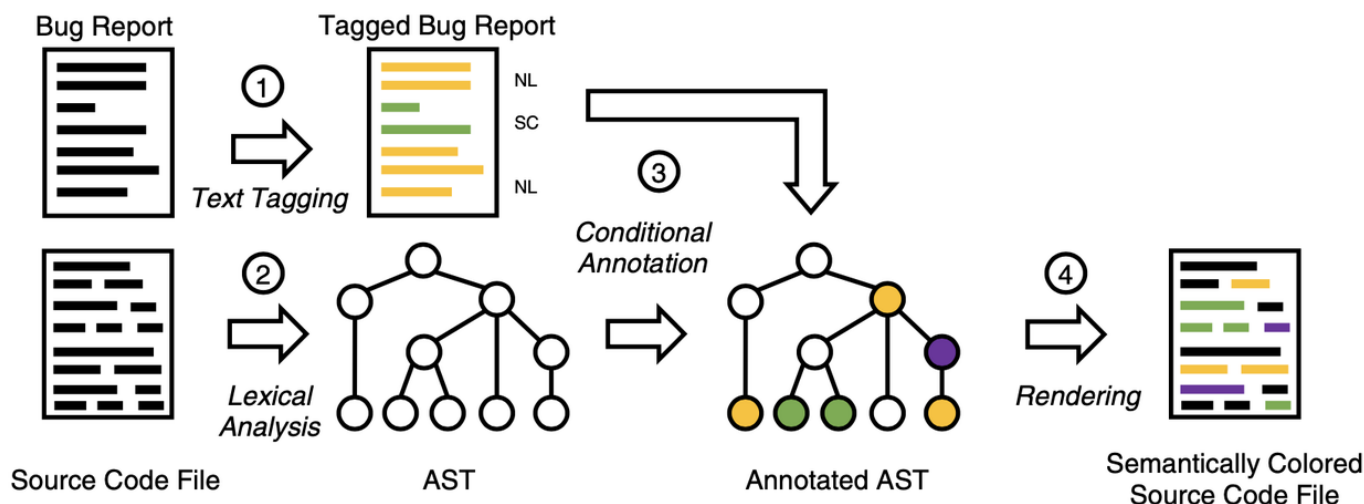
从直觉上来说，stack trace是bug定位的有用信息，但是从作者的研究数据来看，删去报告中的SC，使用NL+ST的效果似乎没有比单纯使用NL好到哪里去，更不如NL+SC。

我的理解是：可以尝试使用ST中的**本地文件**，粗粒度地划分相关和不相关源文件，再使用NL+SC的相似度计算来修正结果

4. 代码高亮算法

基于NL+SC，我个人认为三阶段可以尝试做一下

1. 使用**bug report tagging**对bug报告打标签，NL标记为黄色，SC标记为绿色
2. 词法分析，生成AST（重点）



3. 对AST上的每个节点，节点的值简记为val
如果val出现在NL段中，标记为黄色
如果val出现在SC中，标记为绿色
如果两个都出现，标记为紫色
4. 最后，根据标记后的AST，渲染原来的源代码文本

界面效果（Web App）：

Color coding help: 1 Natural language is highlighted like this 2 Source code is highlighted like this 3 Natural language and source code matches 4 Code lines containing matching text (either natural language or source code) are marked like this ==> 1234

Bug Summary

[JBSEAM-4814] MAX_ROWS reached error message instead displays MAX_COLUMNS

Bug Description 2 Bug Report

When the 65535 row limit of a spreadsheet is reached, the generated exception's message mistakenly displays the limit, as 255.

org/jboss/seam/excel/jxl/JXExcelWorkbook.java Line 130:

```
129 if (currentRowIndex >= MAX_ROWS) {
130     throw new
ExcelWorkbookException(Interpolator.instance()
131         .interpolate("Excel only supports
{0}
rows", MAX_COLUMNS));
132 }
```

JXExcelWorkbook.java

```
org/jboss/seam/excel/jxl/JXExcelWorkbook.java
==> 125     log.trace("Moving from row #0 to #1", currentRowIndex,
126         currentRowIndex + 1);
127 }
==> 128     currentRowIndex++;
129     if (currentRowIndex >= MAX_ROWS) {
==> 130         throw new ExcelWorkbookException(Interpolator.instance()
131             .interpolate("Excel only supports {0} rows", MAX_COLUMNS));
132     }
133 }
134
135 /**
136  * Moves the internal column pointer to the next column, called by the tag
137  * to indicate that a new column has been started. If the pointer exceeds
138  * the maximum allowed, throws an exception. Resets the styles and row
139  * indexes etc.
140  */
141
142 public void nextColumn() {
143     if (log.isTraceEnabled()) {
144         log.trace("Moving from column #0 to #1", currentColumnIndex,
145             currentColumnIndex + 1);
146     }
147     currentColumnIndex++;
==> 148     if (currentColumnIndex > MAX_COLUMNS) {
==> 149         throw new ExcelWorkbookException(Interpolator.instance()
==> 150             .interpolate("Excel doesn't support more than {0} columns",
151                 MAX_COLUMNS));
152     }
==> 153     if (currentRowIndex > maxRowIndex) {
==> 154         maxRowIndex = currentRowIndex;
155     }
==> 156     currentRowIndex = startRowIndex;
157 }
158
159 /**
```

4 Source Code View

Top Source Code Files

1. JXExcelWorkbook.java org/jboss/seam/excel/jxl/JXExcelWorkbook.java
2. JXLHelper.java org/jboss/seam/excel/jxl/JXLHelper.java
3. CsvExcelWorkbook.java org/jboss/seam/excel/csv/CsvExcelWorkbook.java
4. JXLFactory.java org/jboss/seam/excel/jxl/JXLFactory.java
5. UIWorksheet.java org/jboss/seam/excel/ui/UIWorksheet.java

3 Top-5 Ranked Files