软件分析测试第一次作业

1. 研究问题

构造某一语言程序(本作业针对java 语言)AST，计算任意两棵抽 象语法树的相似度，并以此为基础，分析某一项目特定模块的work flow之中的AST变化情况，具体来说，包括以下内容：

1. RQ1.连续的Commit间，模块AST的变化可能有多大？
2. RQ2.连续的发布版本间，模块AST的变化可能有多大？

Bug-fix版本间，模块AST的变化可能有多大？

功能版本间，模块AST的变化可能有多大？

1. RQ3.在什么情况下，相近的两次Commit会使得模块AST发生大的变化？
2. RQ4.开发过程中，模块AST有没有可能发生连续的大的变化？
3. 评估准则
4. 调整同类型节点代码顺序应当不影响AST相似度
5. 改变字面量具体的值应当不影响AST相似度

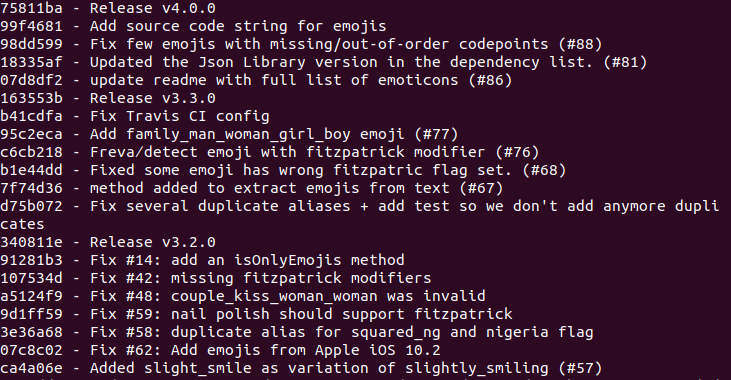
……

1. 实验对象

实验对象来自于开源项目：emoji-java

项目地址：<https://github.com/vdurmont/emoji-java>

Commit 历史概览



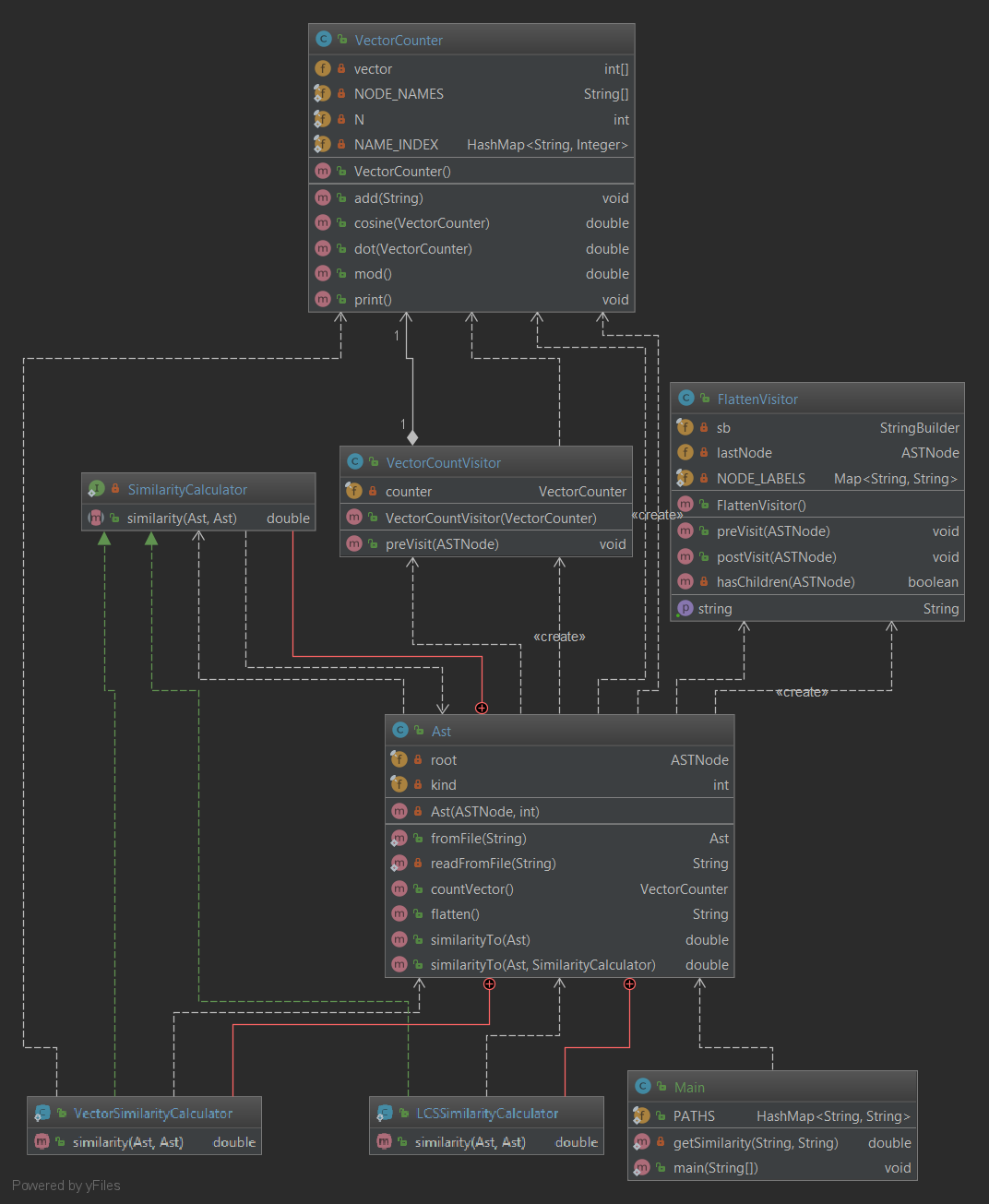
版本历史概览



1. 实验步骤

本实验采用Java作为主要的开发语言

程序UML图如下：



1. AST 构建

AST的构建工作主要借助于

org.eclipse.jdt.core.dom.AST

org.eclipse.jdt.core.dom.ASTNode

org.eclipse.jdt.core.dom.ASTParser

来完成

1. AST之间相似度算法设计

将AST 转化为字符串形式，之后利用LCS算法计算其相似度

Similarity = Length( LCS ) / MAX( Length( AST1 ), Length( AST2 ) )

模块AST间相似度计算方法：

以文件之间AST相似度计算为基础，每个文件的权重为1 ( 因为我们不能确定各个文件之间的重要程度，所以假设为1 )，取平均：

ModuleSimilarity = ( Similarity( file ) ) / Count(files)

1. 连续commit 间，文件及模块AST变化计算

简单来说，通过git log， git checkout 以及相关shell命令完成不同commit id之间的模块内单个代码文本文件之间的比较，而后汇总模块内文件相似度，计算连续commit之间模块级别的相似度。

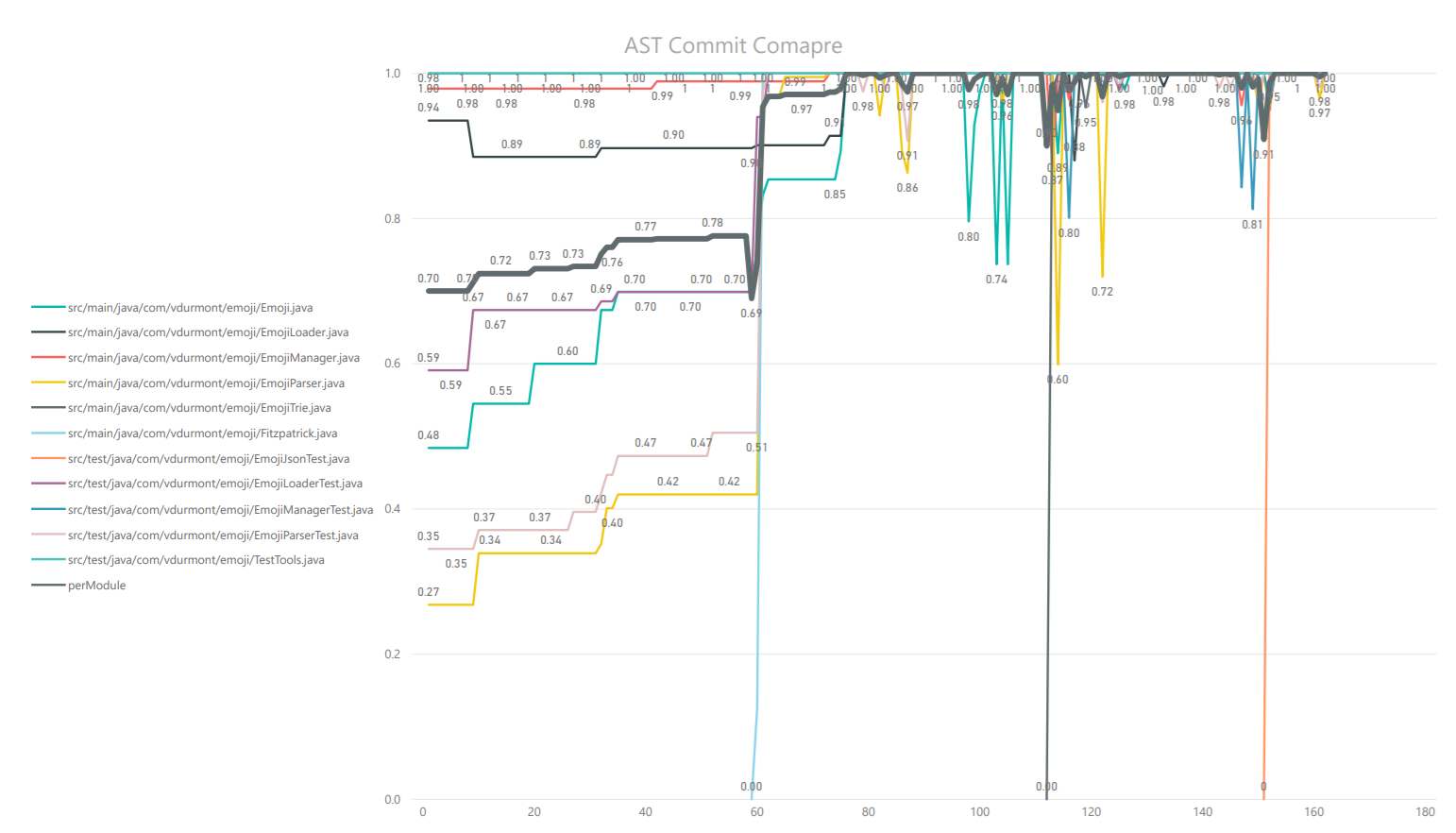
1. 连续发布版本间，文件及模块AST变化计算

同c中的操作, 切换不同的版本，而后再具体得到所需结果。

1. 结果描述与分析

见第5小节

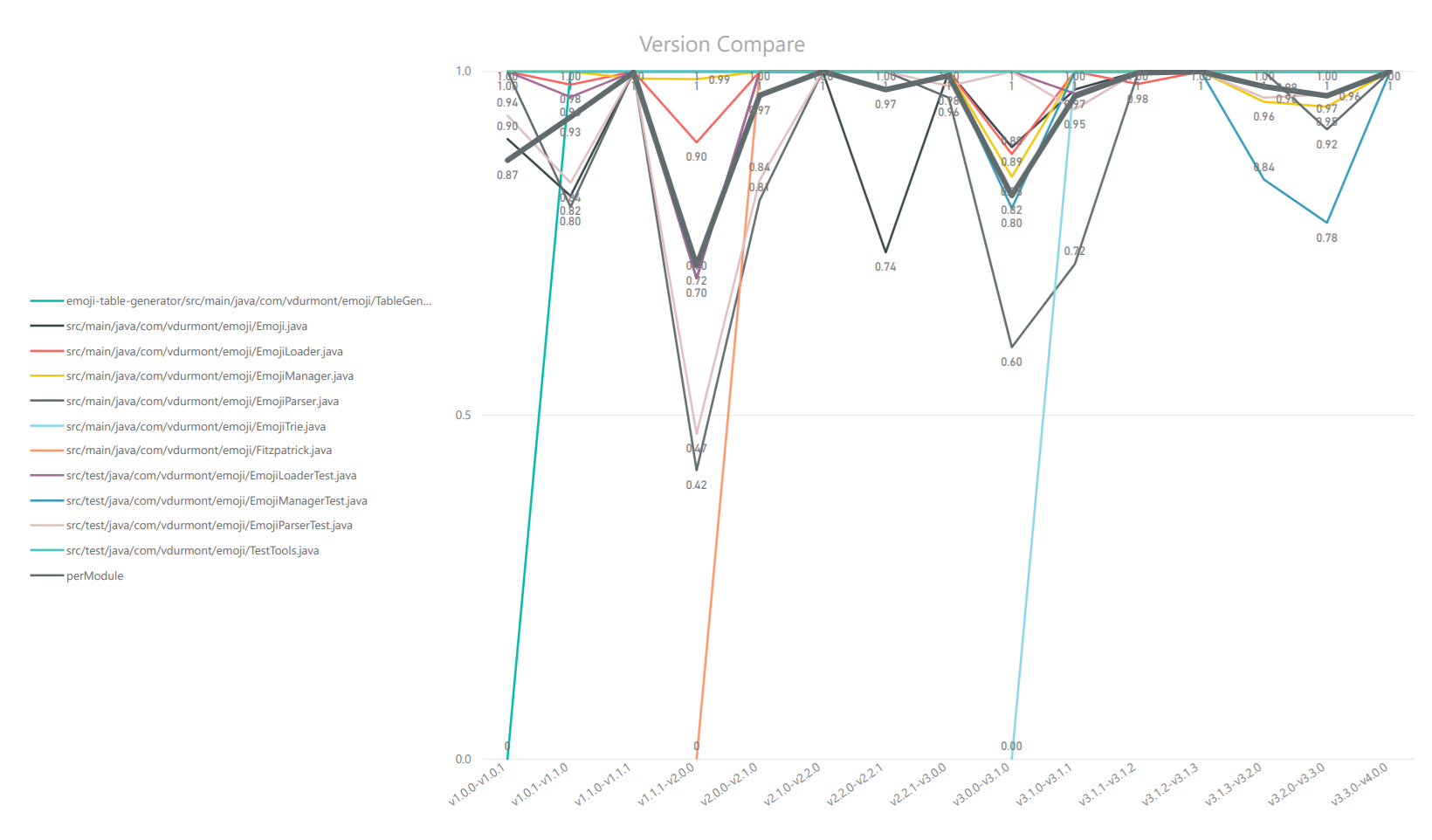
1. 结果呈现与分析
2. RQ1.连续的Commit间，模块AST的变化可能有多大？



连续的Commit间，模块AST的变化

图5-1

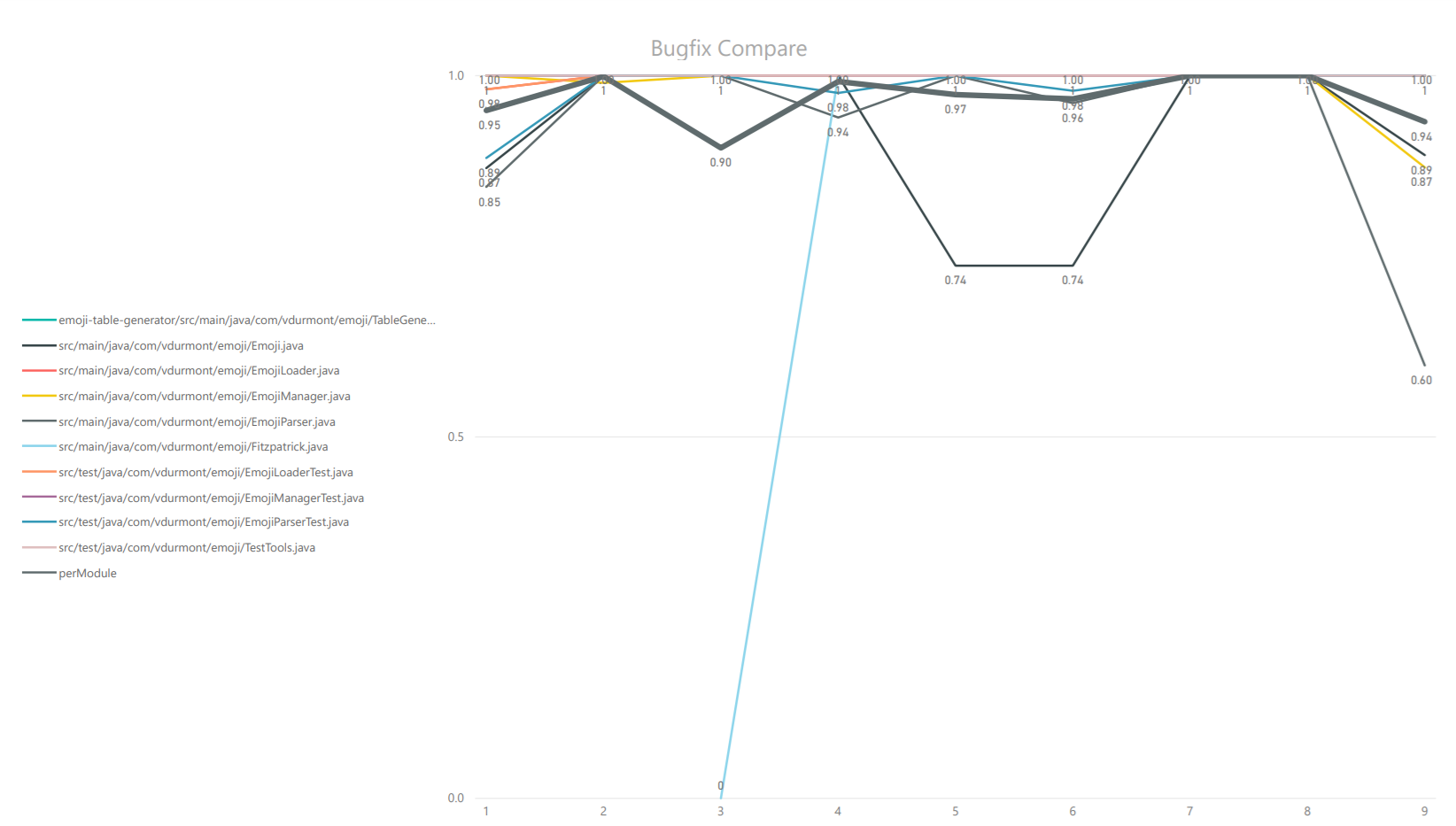
* 1. RQ2.a连续的发布版本间，模块AST的变化可能有多大？



连续的发布版本间，模块AST的变化

图5-2

* 1. RQ2.b bugfix 之间，模块AST的变化可能有多大？



Bug-fix，模块AST的变化

图5-3

1. RQ3.在什么情况下，相近的两次Commit会使得模块AST发生大的变化？

由图5-1可知，早期开发阶段，模块AST变化较大，初步推断是因为早期模块尚处于非稳定阶段，功能增减和bug修复比较频繁，另外受本实验模块similarity计算策略的影响，当在连续两次commit之间有代码文件新建时，模块AST变化较大。

1. RQ4.开发过程中，模块AST有没有可能发生连续的大的变化？

由我们的实验结果即图5-1可知，在模块开发早期阶段，有可能。

1. 小结

通过本次实验，了解了AST的具体构建过程以及对如何计算AST之间相似度有了一定的认识。

附：项目地址 <https://github.com/wen-zheng/SAT-1>

REF:

https://github.com/nettee/ASTComparator