南京大学
NANJING UNIVERSITY

软件工程课程实验介绍

SEG
Nanjing University

2022/9/8

实验内容

- 设计实现一个程序等价性确认工具
  - 熟悉程序等价的定义和确认方法
  - 自动判断两个程序是否等价
  - 提供交互式确认的接口
  - 对实现的程序等价性确认工具进行测试

实验大纲

- 实验一：人工确认程序等价性
- 实验二：软件需求分析
- 实验三：软件设计
- 实验四：软件实现
- 实验五：交互式确认方法
- 实验六：软件分析与测试

实验一：人工确认程序等价性

- 现有工具 eqminer[1]，可以根据给定的源程序，生成若干代码块（每个代码块都被包装成一个函数），并通过动态执行这些函数来将其聚类成若干的 cluster，每个 cluster 内的函数被认为是相互等价的。
- eqminer 判断等价的原理是比较两个程序在输入相同的时候，是否具有相同的输出。
- 显然，通过动态执行后生成的 cluster 内的所有函数并不一定就是完全相互等价的，因为动态执行的过程是有穷尽的。此时就需要人工确认来保证其等价性。

[1] Jiang L, Su Z. Automatic mining of functionally equivalent code fragments via random testing[C]//Proceedings of the eighteenth international symposium on Software testing and analysis. 2009: 81-92.

实验一：人工确认程序等价性

- 第一次实验给每位同学分配了 eqminer 生成的 cluster，同学们通过确认每个 cluster 内各函数之间的等价性，熟悉一下等价判断的过程，了解eqminer判断等价的逻辑和效果。

实验内容

- 每个人会分得一组数据，每组数据中含有若干文件夹（每个文件夹即为一个cluster）。
- 对于每个文件夹（文件夹中含有 n 个文件），判断其中任意2个文件的等价性，并记录在 csv 文件中。
  - 若等价，则记录在groupID-equal-pairs.csv中；
  - 若不等价，则记录在groupID-inequal-pairs.csv中。
- 每个文件夹产生的总数据量（包括等价和不等价）应为 C_n^2 。
  - 在没有重复数据的情况下。重复数据的情况处理下文会提到。

如何判断程序等价性

- 以__dyc_read开头的函数表示读取不同格式的输入。下图中，红框中的内容即为该函数的输入。

```
{
ut_dbg_null_ptr = __dyc_read_ptr__typdef_ulint();
dict_sys = __dyc_read_ptr__typdef_dict_sys_t();
name_len = (ulint )__dyc_readpre_byte();
is_sys_table = (ulint )__dyc_readpre_byte();
table = __dyc_read_ptr__typdef_dict_table_t();
__dyc_funcallvar_6 = __dyc_read_ptr__typdef_dtuple_t();
__dyc_funcallvar_7 = __dyc_read_ptr__typdef_dfield_t();
__dyc_funcallvar_8 = __dyc_read_ptr__void();
__dyc_funcallvar_9 = (ulint )__dyc_readpre_byte();
__dyc_funcallvar_10 = __dyc_read_ptr__typdef_rec_t();
__dyc_funcallvar_11 = (unsigned char *)__dyc_read_ptr__char();
__dyc_funcallvar_12 = __dyc_readpre_byte();
__dyc_funcallvar_13 = (ulint )__dyc_readpre_byte();
__dyc_funcallvar_14 = (unsigned char *)__dyc_read_ptr__char();
__dyc_funcallvar_15 = __dyc_read_comp_100dulint_struct();
__dyc_funcallvar_16 = __dyc_read_ptr__typdef_dict_field_t();
__dyc_funcallvar_17 = __dyc_read_ptr__typdef_dict_col_t();
__dyc_funcallvar_18 = __dyc_readpre_byte();
```

如何判断程序等价性

- 以__dyc_print开头的函数表示输出不同格式的数据，其参数即为函数输出。下图中，红框中的内容即为该函数的输出。

```
__dyc_print_ptr__typdef_ulint(ut_dbg_null_ptr);
__dyc_print_ptr__typdef_dtuple_t(tuple);
__dyc_print_ptr__typdef_dfield_t(dfield);
__dyc_print_ptr__typdef_rec_t(rec);
__dyc_print_ptr__char(field);
__dyc_print_ptr__char(name_buf);
__dyc_printpre_byte(type);
__dyc_printpre_byte(space);
__dyc_printpre_byte(n_fields);
__dyc_print_ptr__char(buf);
__dyc_print_comp_100dulint_struct(id);
__dyc_print_ptr__typdef_dict_field_t(tmp___7);
__dyc_print_ptr__typdef_dict_col_t(tmp___8);
__dyc_printpre_byte(tmp___10);
__dyc_print_ptr__typdef_dict_field_t(tmp___13);
__dyc_print_ptr__typdef_dict_col_t(tmp___14);
__dyc_printpre_byte(tmp___16);
}
```

如何判断程序等价性

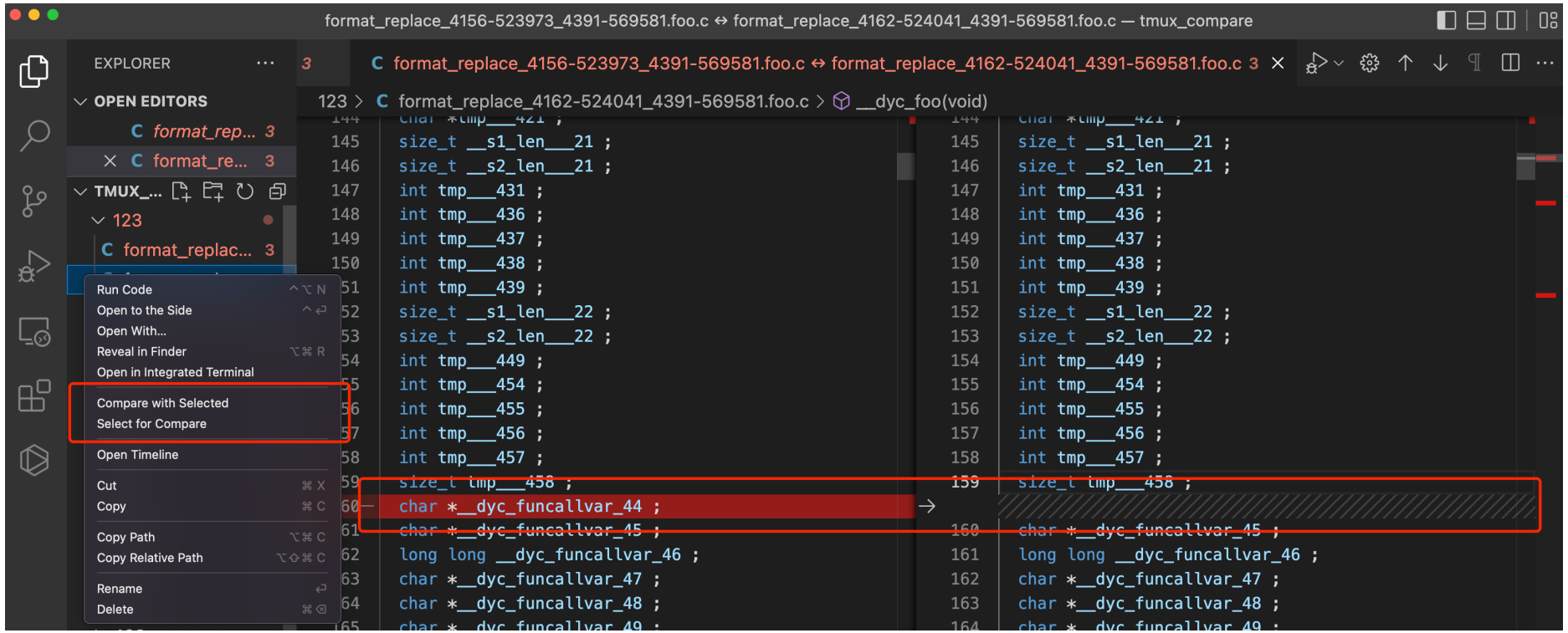
- 函数的所有输入值都是随机生成的，且相互独立。
- 判断等价时两个函数的输入（或输出）个数可以不同，顺序也可以不同。
- 将输入和输出都看作【集合】的概念，比较两个函数在输入相同的时候，输出值的集合是不是呈现包含关系。若呈现包含关系，则二者等价，否则为不等价。
  - 当输入个数不同时（例如一个为 i，一个为 j，且 i<j），输入相同即表示两个函数拥有相同的 i个输入，且输入个数为 j 的那个函数会有额外的 j-i 个随机生成的输入。输入值也是集合的概念，两个函数中相同的变量名不一定具有相同的输入值。

如何判断程序等价性

- 另一种等价的情况是待判断的两个函数都遇到运行时错误或都陷入无限循环。
- 关于重复数据： 注意，当两个文件完全相同时，忽略其中一个。例如若某文件夹中原本包含5个文件，但其中有2个文件相同，则最后生成的数据量应为 $C_4^2=6$。

一些建议

- 可以从 diff 命令开始，找出两个函数的主要差异，再基于这些差异判断等价与否（或使用 vscode 的 compare 功能）。如果发现两个文件的diff差异不涉及输入输出，那就可以直接认为是等价的了。



一些建议

- 注意一些 goto 语句。例如下图中的第356行会导致从357行之后的逻辑不会被执行到，直接跳到程序输出。
- 一般来说不需去理解程序！

```
352     tmp___60 = 0;
353     tmp___61 = 0;
354   #line 521
355     yystate = yyn;
356     goto __dyc_dummy_label;
357     yydefault:
358   #line 527
```

实验提交

- 截止日期：2022.9.22 24:00
- 提交方式：将两个 csv 文件，分别存放等价的程序对和不等价的程序对(csv 文件的具体格式详见课程网站上示例)，打包成 zip 文件提交到602114946@qq.com，提交格式为：

```
lab1-学号-姓名.zip
  └lab1-学号-姓名
      ├groupID-equal-pairs.csv
      └groupID-inequal-pairs.csv
```

- QQ群：831918103
- https://seg.nju.edu.cn/curriculums/Software_Engineering_(Fall_2022)