

CS 562

Date: 3.10.16

Ruizhi Guo

Final Report of CS 562 Project

Description of my work

This project is using the Template Scripting Testing Language (short with TSTL as follow) to test a third party library. In this testing assignment, the name of the Python library which I would like to use TSTL to test is Xpinyin. The source code of this library can be download from the Github website and the link come as follow <https://github.com/lxneng/xpinyin>. This library is developed by Eric Lo, and he is come from China. This python library is use for transform the Chinese word (writing character) in to Chinese pinyin (the pronunciations of the word). The reason why I choose this library is it can help user get the pinyin for the input(the writing word) from user. With the developing of technology, young people rely on the cell phone and computer. So the ability of recognize the pronunciation of word is become weak. Through this library the user input the word then they would get the pronunciation of the word from the output.

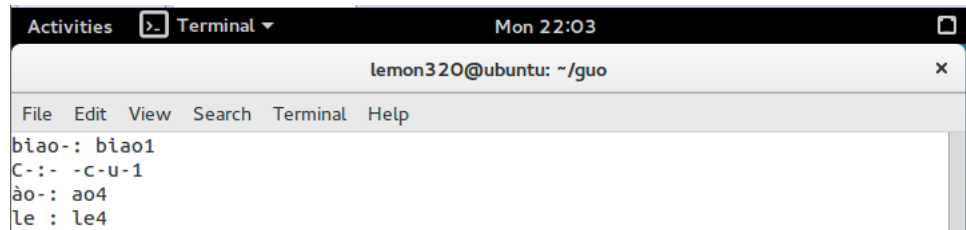
First I found this python library in the Github website, it is a third party library, so there are some small problems to install it on mac. But it has no problem to install on windows. I would like try to debug this python library from these aspects, multiple pronunciation, unusual Chinese word, and complex long sentence. In order to debug for this library as much as possible, I would like to design more test data for it. The first kind test data is multiple

pronunciation. In Chinese language, some word may have different pronunciation in different situation. In order to test this problem for this python library, I would like to input a sentence include this multiple pronunciation twice or more times. Then check the output to check if the pronunciation is right under the situation. Such like Chinese word “强”, this word has three pronunciations and different meanings. Second case is unusual Chinese word. In this case, I need to collect enough unusual word as the input to test the python library. If the library can not output the right pronunciation for the unusual word that may mean the support data for this library is not strong enough. The third case is complex long sentence. In this problem, I combine two aspects one is the longest strings this library can accept the other is the long string is includes the multiple pronunciation. So if I just set a paragraph of Chinese word as the input, then check the output.

The bug I Found

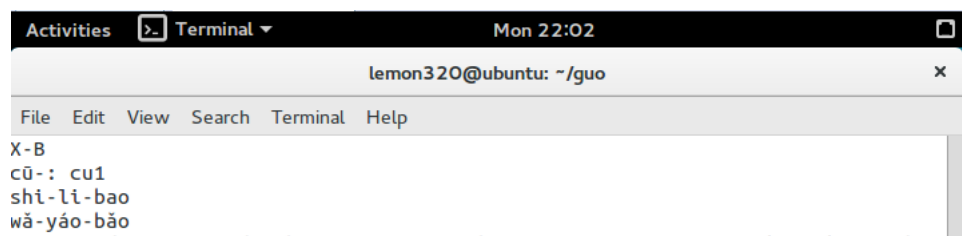
In my opinion that this library is an interesting design and it would be very useful for some Chinese teaching program. As we know that Chinese is hard to learn, so design such a library is even harder. It need to consider a lot of different situations. So in my debug part, I try from three different parts to find bugs and they are unusual Chinese word, multiple pronunciation and long complex sentence. In order to find bug as much as possible, I would like to design some different test data for it. Then I find some bug as I expect in the proposal.

The first kind of bug is the unusual Chinese word. I try to find some hard Chinese word on the dictionary and set them in the input. The library could successfully recognize the input, but the problem is the output part. The pronunciation for these word is not correct. For example, my input is “麤 cu1”, the output is not correct.

A terminal window titled 'Terminal' with a timestamp of 'Mon 22:03'. The user is 'lemon320@ubuntu' in the directory '~/guo'. The terminal shows a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. Below the menu, the following text is displayed:

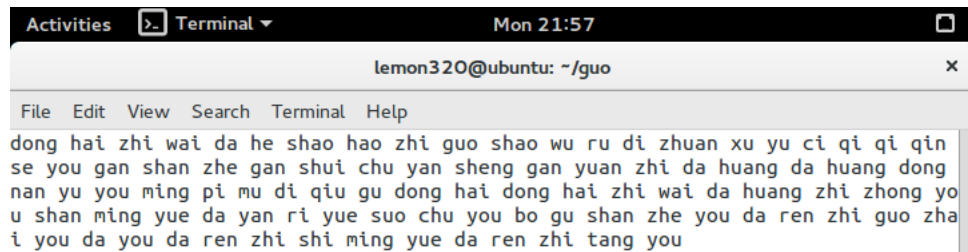
```
biao-: biao1
C-:- -C-U-1
ào-: ao4
le : le4
```

The second kind of bug is the multiple pronunciation. In this part, I set some multiple pronunciation Chinese word in the input. As we know that some Chinese words under different situation would have different pronunciation. The output is not as good as I expect. The library only provides one pronunciation for the multiple pronunciation word. So it can not discover the multiple words and provide the correct pronunciation for them. For example, my input is “瓦窑堡 wayaobu” the output only provide one pronunciation for each multiple pronunciation word.

A terminal window titled 'Terminal' with a timestamp of 'Mon 22:02'. The user is 'lemon320@ubuntu' in the directory '~/guo'. The terminal shows a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. Below the menu, the following text is displayed:

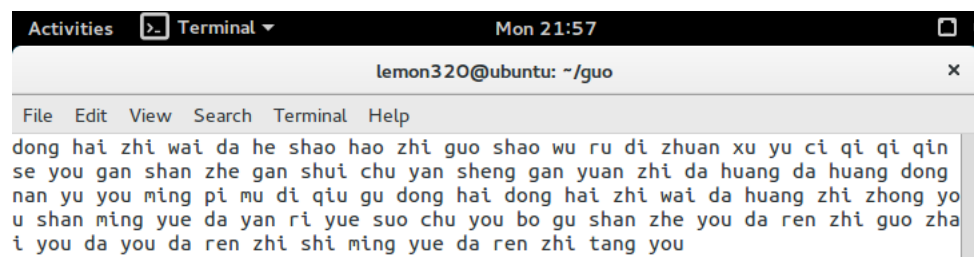
```
X-B
cū-: cu1
shì-lì-bao
wǎ-yáo-bǎo
```

In the third part, the complex long sentence, it contains both bug from the above situations. In the complex long sentence case, I also have three different tests. The first case is normal complex long sentence. In this test, the above error still happen.



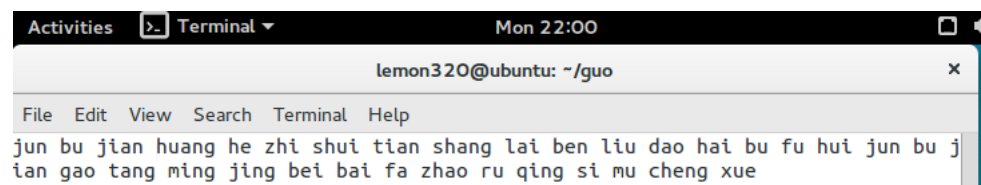
```
lemon320@ubuntu: ~/guo
File Edit View Search Terminal Help
dong hai zhi wai da he shao hao zhi guo shao wu ru di zhuan xu yu ci qi qi qin
se you gan shan zhe gan shui chu yan sheng gan yuan zhi da huang da huang dong
nan yu you ming pi mu di qiu gu dong hai dong hai zhi wai da huang zhi zhong yo
u shan ming yue da yan ri yue suo chu you bo gu shan zhe you da ren zhi guo zha
i you da you da ren zhi shi ming yue da ren zhi tang you
```

The second case is some old Chinese poem. In this test, I find a really long poem and try to discover the max length input. In this case, there is no error happen.



```
lemon320@ubuntu: ~/guo
File Edit View Search Terminal Help
dong hai zhi wai da he shao hao zhi guo shao wu ru di zhuan xu yu ci qi qi qin
se you gan shan zhe gan shui chu yan sheng gan yuan zhi da huang da huang dong
nan yu you ming pi mu di qiu gu dong hai dong hai zhi wai da huang zhi zhong yo
u shan ming yue da yan ri yue suo chu you bo gu shan zhe you da ren zhi guo zha
i you da you da ren zhi shi ming yue da ren zhi tang you
```

The third case is traditional Chinese characters. In this case, I discover a different form of Chinese characters try to find out if this library supports it. The result is that this library is perfect enough that the designer already covered this part.



```
lemon320@ubuntu: ~/guo
File Edit View Search Terminal Help
jun bu jian huang he zhi shui tian shang lai ben liu dao hai bu fu hui jun bu j
ian gao tang ming jing bei bai fa zhao ru qing si mu cheng xue
```

Tester works

After one term of use this tester that I think this tester is good enough for recommended to others. This tester could tell if the code is wrong after the code is passed. Another point is the randomtester, with it help I could easily find the bug, because it could provide random strings. After I use TSTL test Pinyin library, I think this library is good. The designer cover most of cases could happen. Although I find some bugs in this library, I think this library can be improved. In the future work, the designer need to consider the multiple pronunciation problem.

Coverage

As we know that coverage is one of the important standard that to evaluate out work. In this time my code coverage is 75.25%. In the middle of this term the code coverage is 61.31%.

```
STOPPING TEST DUE TO TIMEOUT, TERMINATED AT LENGTH 25
STOPPING TESTING DUE TO TIMEOUT
75.2577319588 PERCENT COVERED
6.17230510712 TOTAL RUNTIME
12 EXECUTED
1125 TOTAL TEST OPERATIONS
6.02420687675 TIME SPENT EXECUTING TEST OPERATIONS
0.0495181083679 TIME SPENT EVALUATING GUARDS AND CHOOSING ACTIONS
0.00192952156067 TIME SPENT CHECKING PROPERTIES
6.02613639832 TOTAL TIME SPENT RUNNING SUT
0.0326759815216 TIME SPENT RESTARTING
0.0 TIME SPENT REDUCING TEST CASES
69 BRANCHES COVERED
52 STATEMENTS COVERED
```

TSTL's Good and Bad

After one term with TSTL, one of the good points I find is TSTL could provides an easy way to create a lot of random data to test the object. The professor also mention this in the lecture that before we use TSTL we have to create different cases random test, sometimes the test data maybe not really "random". TSTL creates a easy way to create really random(compare with by hand) test for the project. In my opinion, this is the benefit of TSTL. Another good point

that I think is TSTL is easy to learn and use. It is new way to do debug and it could find some bug that other way can not find. The logic of TSTL is easy to understand and the grammar of TSTL is not hard for new user.

On the other side, TSTL is not the perfect test tool. After I use it in this term, I find that it still has some disadvantages. One of them relate to my project is TSTL is not support other language(non English language) very well. In this part, I have to type (# -*- coding: utf-8 -*-") every time after I create a new sut.py.

Summary

After this term I learn that how to use TSTL. I find that TSTL is very friendly to new user and it provides a great help to user in debug. Although it has some problems with support non English, I believed it could be fixed in the future and I would like to do some test with TSTL in the future work.