CS562: Applied Software Engineering

Part 3: Bug/progress report

Hafed Alghamdi
Email: Alghamha@oregonstate.edu

- **Describe some bugs you found**

Testing Astropy.table library using TSTL revealed some unexpected bugs. These bugs varies in criticality from bad API to serious bugs that should not exist in such library. The following is a summary of these discovered bugs:

1- ***Updating Columns' Unit Field Bug:***
This bug occurs when inserting a column at least twice to update its values. The library will update the values (tuples) of a column but the unit field will not get the new updates. This might cause a devastating damage to the table's data when for example a user has a loop to remove the tables based on their units' values. Based on user's assumption that the update should be implemented and further actions can be applied. As a result, this will eventually results in data inconsistency. Someone, could argue that this might be a part of the design. However, this kind of argument cannot be taken into account and this can be considered as a bad design because the designer should expect users' unexpected behaviors. Therefore, a good design could be, either updating all the values including the unit field or not updating the values at all and raising an exception error or at least documenting this kind of action is not accepted and may results in data inconsistency.

2- ***Local variable 'new_name' referenced before assignment:***
This exception occurs when inserting a column randomly to the table while the rename_duplicate option is enabled and the column is not exist in the table. This bug is impractical as it shows some API weaknesses that may lower user's overall satisfaction. Usually, user's expectations when adding a column to a table with the option rename-duplicate enabled, the column will be added no matter what, and even if it is not exist in the table it will be at least inserted with the same name. A proper action for a such situation is checking for the column existence first and based on that apply an action such as inserting the column with the same name if it is not exist otherwise rename_duplicate when the column is exist.

3- ***Updating Tuples (Indices) of Type (String) Bug:***
This bug occurs when updating a tuple value of type (string). In Astropy.table library, there are different types of columns including Integers, float, string, object, etc. String type was interesting to test because it uses the length of the first insertion of columns' tuples and hard coded the longest string length as a type. For example, if you have 'Hafed' as the longest string tuple, then the corresponding type of column would be S5. This was an interesting technique to test and a bug was detected when updating the tuples after normal columns insertion. The detected bug appears when updating a tuple with longer string after normal column insertion where the updated tuple will only have a portion of the new string. Apparently, it is only updating the field based on the original column type when it was first inserted, ignoring that it can be updated with longer strings which in turn results in table inconsistency and unreliable

data. This is a major bug because the user is expecting the field to be updated with the new string value but in reality it is not, because of this column type string specification.

- **Explain progress to date**

  Eighty percent of the hardest part of testing is almost done. There are many functions have been tested as follows:

  1. Inserting and deleting columns
  2. Adding and deleting rows
  3. Astropy.table vs python built-in sorting
  4. Updating tuples (columns values)
  5. Columns options, properties

- **Estimate your progress by end of term**

  In addition to the final report, there are some functions still have not been tested yet. However, these functions will not be as hard as what's been achieved so far. These function are as follows:

  1. as_array function which returns a copy of the table as np
  2. copy the table Function
  3. Return Index_Column function
  4. Reverse Table values function

- **Discuss quality of the SUT**

  Astropy table library is very large and complicated but it is still very useful and widely used. However, I believe the developers and designers of this library considered end-users awareness of their data but did not take into account end-user misuse or using the library for huge amount of data that little misbehaviors may results in data inconsistency. Most of the functions are very well tested and running as required, and the number of errors detected is very small comparing to the library size which in turn shows some sort of high quality. I believe this library is well done but it will be perfect if the developers take into account end-users misuses.

- **Code Coverage:**

  The test coverage of the Astropy code is almost 23 percent so far according to python coverage report. However, most of the highly used functions have been tested as aforementioned and I believe the low coverage percent was a result of the untested parts related to printing, importing files and exporting files functions that has many properties, branches and functions that were not and won't be tested in this project.

  Note: TSTL coverage worked for me using the normal import but it was not giving any kind of accuracy. However, when using 'source: <path> to the library code, because it cannot be copied to the TSTL folder' it worked and gave better results.

References:

1- http://www.astropy.org/
2- http://docs.astropy.org/en/stable/install.html
3- http://docs.astropy.org/en/stable/index.html
4- http://docs.astropy.org/en/stable/table/index.html#module-astropy.table
5- http://docs.astropy.org/en/stable/index.html
6- http://docs.astropy.org/en/stable/units/index.html
7- http://docs.astropy.org/en/stable/index.html
8- https://github.com/astropy/astropy
9- https://github.com/agroce/tstl
10- http://docs.astropy.org/en/stable/table/modify_table.html
11- https://docs.python.org/2/tutorial/index.html