Verification and Validation Report for Centrality In Graphs

Atiyeh Sayadi

April 18, 2024

1 Revision History

Date	Version	Notes
April 10, 2024	Notes	

2 Symbols, Abbreviations and Acronyms

symbol	**description**
T	Test

Contents

List of Tables

List of Figures

This document provides a summary of the tests conducted for the centrality project in graphs.

3 Functional Requirements Evaluation

This section tests the functional requirements of the system, the description of which is as follows:

1. Test1: Bound Testing for degree centrality
   Based on requirements 1 and 2 of this project, it is necessary for the degree centrality to be accurately calculated for all nodes. We know that this metric for each node is between zero and one. Therefore, a function needs to be written to evaluate this value for each node based on the allowable range and report any discrepancies found.

2. Test2: Bound Testing for closeness centrality
   Based on requirements 3 and 4 of this project, it is necessary for the closeness centrality to be accurately calculated for all nodes. We know that this metric for each node is between zero and one. Therefore, a function needs to be written to evaluate this value for each node based on the allowable range and report any discrepancies found.

The implementation of both tests is available in the (test_cig).

4 Nonfunctional Requirements Evaluation

4.1 Usability

No user feedback has been obtained for this test.

4.2 maintainability

1. Test3: Code Testing
   By utilizing the Python's Pylint tool, the codes can be standardized, ensuring high maintainability.

4.3 Accuracy

1. Test4: Correctness Testing for DC

   Using the NetworkX library functions in Python, the centrality of each node is calculated for the given graph, and it is compared with the output of the program(Based on requirements 1 and 2).

   How test will be performed: Nodes for which the calculated centrality differs from the library function output are displayed.

2. Test5: Correctness Testing for CC

   Using the NetworkX library functions in Python, the centrality of each node is calculated for the given graph, and it is compared with the output of the program(Based on requirements 3 and 4).

   How test will be performed: Nodes for which the calculated centrality differs from the library function output are displayed.

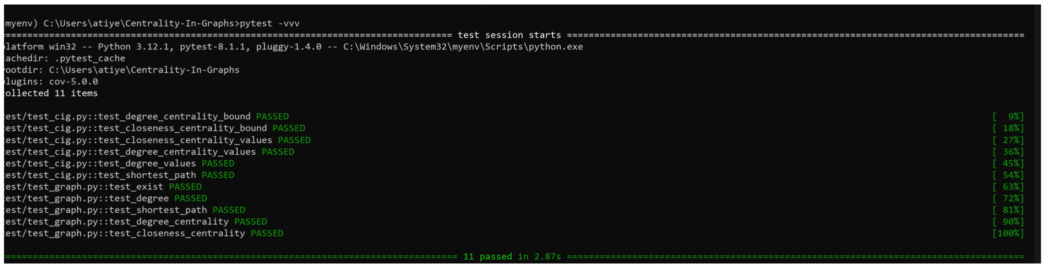
The implementation of both tests is available in the (test_cig).

5 Comparison to Existing Implementation

Previously, the plan was to implement the main graph of the project in the form of an adjacency matrix, representing the relationships between nodes, or the adjacency matrix, and related concepts. However, following feedback from Dr. Spencer Smith, it was decided to design this project in a modular manner based on the abstract data type, which is the graph itself. This modular design approach significantly enhances usability and maintainability.

6 Unit Testing

The implementation of these tests is available in the (test_graph).



```
myenv) C:\Users\atiye\Centrality-In-Graphs>pytest -vvv
================================================================== test session starts ==================================================================
latform win32 -- Python 3.12.1, pytest-8.1.1, pluggy-1.4.0 -- C:\Windows\System32\myenv\Scripts\python.exe
achedir: .pytest_cache
ootdir: C:\Users\atiye\Centrality-In-Graphs
lugins: cov-5.0.0
ollected 11 items

est/test_cig.py::test_degree_centrality_bound PASSED                                                                                             [  9%]
est/test_cig.py::test_closeness_centrality_bound PASSED                                                                                          [ 18%]
est/test_cig.py::test_closeness_centrality_values PASSED                                                                                         [ 27%]
est/test_cig.py::test_degree_centrality_values PASSED                                                                                            [ 36%]
est/test_cig.py::test_degree_values PASSED                                                                                                       [ 45%]
est/test_cig.py::test_shortest_path PASSED                                                                                                       [ 54%]
est/test_graph.py::test_exist PASSED                                                                                                             [ 63%]
est/test_graph.py::test_degree PASSED                                                                                                            [ 72%]
est/test_graph.py::test_shortest_path PASSED                                                                                                     [ 81%]
est/test_graph.py::test_degree_centrality PASSED                                                                                                 [ 90%]
est/test_graph.py::test_closeness_centrality PASSED                                                                                              [100%]

=================================================================== 11 passed in 2.87s ==================================================================
```

Figure 1: Test logs

7 Changes Due to Testing

After numerous reviews, this code has undergone several changes, including the definition of the graph data type and the class related to graph implementation. Additionally, the method for displaying output to highlight important nodes has been modified per Dr. Spencer Smith's directive.

8 Automated Testing

All tests have been automated using Python's testing framework (Pytest) and linting tool (Pylint).

9 Trace to Requirements

10 Trace to Modules

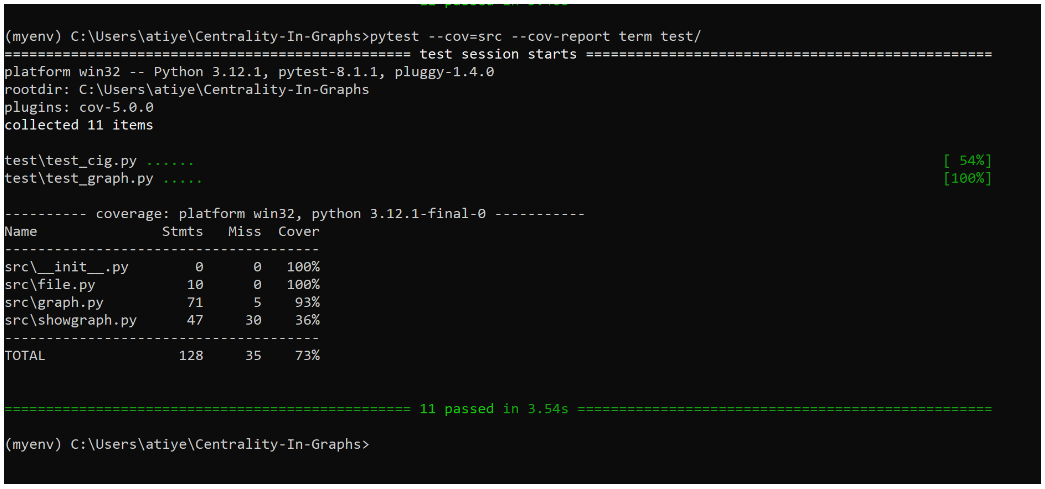
	FR1	FR2	FR3	FR4	NFR1	NFR2	NFR3	NFR4
Bound Testing for degree centrality	X	X			X			
Bound Testing for closeness centrality			X	X	X			
Code Testing					X		X	
Correctness Testing for DC	X	X			X			
Correctness Testing for CC			X	X				
test_degree_centrality	X	X						
test_closeness_centrality			X	X				
test_exist	X	X	X	X				
test_degree	X	X						
test_shortest_path			X	X				

Table 1: Trace to Requirements

	File	Graph	ShowGraph
Bound Testing for degree centrality		X	
Bound Testing for closeness centrality		X	
Code Testing	X	X	X
Correctness Testing for DC		X	X
Correctness Testing for CC		X	X
test_degree_centrality		X	
test_closeness_centrality		X	
test_exist		X	
test_degree		X	
test_shortest_path	X		

Table 2: Trace to Modules

11 Code Coverage Metrics



```
(myenv) C:\Users\atiye\Centrality-In-Graphs>pytest --cov=src --cov-report term test/
================================================ test session starts ================================================
platform win32 -- Python 3.12.1, pytest-8.1.1, pluggy-1.4.0
rootdir: C:\Users\atiye\Centrality-In-Graphs
plugins: cov-5.0.0
collected 11 items

test\test_cig.py ......                                                                                       [ 54%]
test\test_graph.py .....                                                                                      [100%]

---------- coverage: platform win32, python 3.12.1-final-0 -----------
Name                Stmts   Miss  Cover
---------------------------------------
src\__init__.py         0      0   100%
src\file.py            10      0   100%
src\graph.py           71      5    93%
src\showgraph.py       47     30    36%
---------------------------------------
TOTAL                 128     35    73%


================================================ 11 passed in 3.54s =================================================

(myenv) C:\Users\atiye\Centrality-In-Graphs>
```

Figure 2: Code Coverage Metrics