



International Journal of Emerging Technologies in Computational and Applied Sciences (IJETCAS)

www.iasir.net

Estimate Minimum Cost Path Sequence Based on Dynamic Programming Approach

Kulvinder Singh Handa¹, Mehak Kalra²

^{1,2} Department of Computer Engineering,

University Institute of Engineering and Technology,

Kurukshetra University, Kurukshetra, Haryana, INDIA

E-mail(s): kshanda@rediffmail.com, mehakkalra.17@gmail.com

Abstract: Testing plays an important role in the software development to identify the difficulties in process very well. In case of code based testing, path testing plays a very crucial role. The importance of path testing is to check the code coverage as well as to identify the code significance. Better the test sequence is estimated, better the testing is performed. It also affects the testing code and test case reusability. There are number of factors on which code sequence is decided. This paper presents a framework that helps in implementing dynamic programming approach that will work on the basis of fault as well as future aspects of test cases to estimate path sequence.

Keywords: White box testing, Path testing, Dynamic Programming

I. Introduction

Testing is basically a process to detect errors in the software product. It is performed to check the quality for the end client. Its role is to get the requirements and design documents from the solutions architect and functional architect and adapt in the form of test cases and test scripts according to the test plan. All the test cases gets validate thoroughly to check or find the defects in the application or system [1]. Objective of this paper is to generate minimum path sequence by implementing dynamic programming. It works on the basis of fault as well as future aspect of a test case. The cost of each test case based on fault occurrence chances in the code block and future test chances of code during maintenance is determine. Then dynamic programming uses graph based cost estimation for calculating the cost of test cases in path testing.

Dynamic programming refers to simplifying a complicated problem by breaking it down into simpler sub-problems in a recursive manner. This approach seeks to solve each sub problem only once, thus reducing the number of computations. This is especially useful when the number of repeating sub-problems is exponentially large [2]. Two key attributes that a problem must have in order for dynamic programming to be applicable: optimal substructure and overlapping sub-problems. Optimal substructure means that the solution to a given optimization problem can be obtained by the combination of optimal solutions to its sub-problems. Overlapping sub-problems means that the space of sub-problems must be small, that is, any recursive algorithm solving the problem should solve the same sub-problems over and over, rather than generating new sub-problems [2].

White box testing uses the internal mechanism of a system to create test cases, which uses calculated paths to discover unidentified bugs within the system. WBT is a test case design method that uses the control structure of the procedural design to derive test cases [3]. The test cases derived from white box testing methods will:

- Guarantee that all independent paths within a module have been exercised at least ones.
- Exercise all logical decisions on their true and false side.
- Execute all loops at their boundaries and within their operational bounds.
- Exercise internal data structures to ensure their validity [3].

WBT needs to be adopted under unit level testing strategy. It can be adapted to a limited extent under integration testing if situation warrants for it. Basic path testing and control structure testing are some of the most widely used White box testing technique. It is the testing method in which the user will test the application structure how it is acting [4].Classification of White box testing:

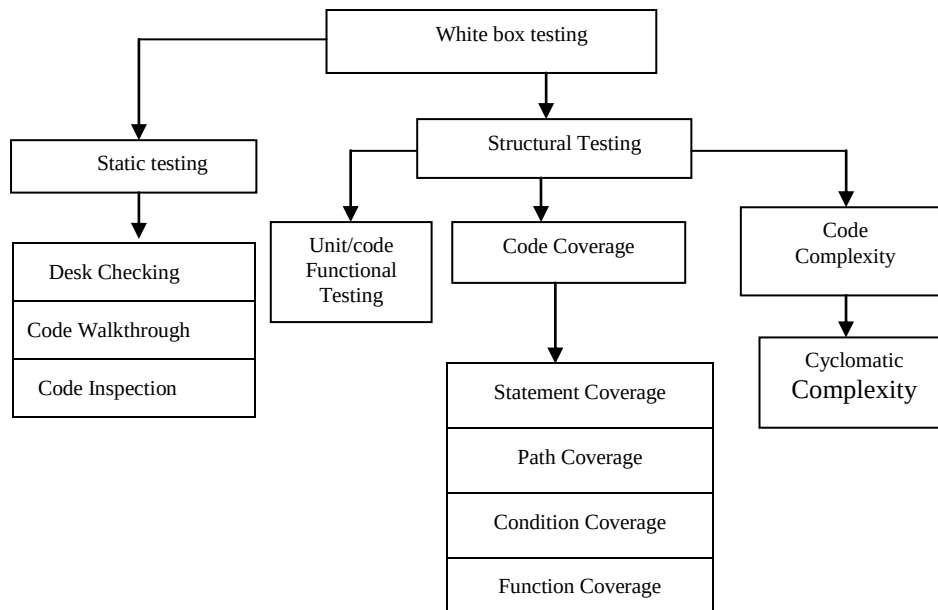
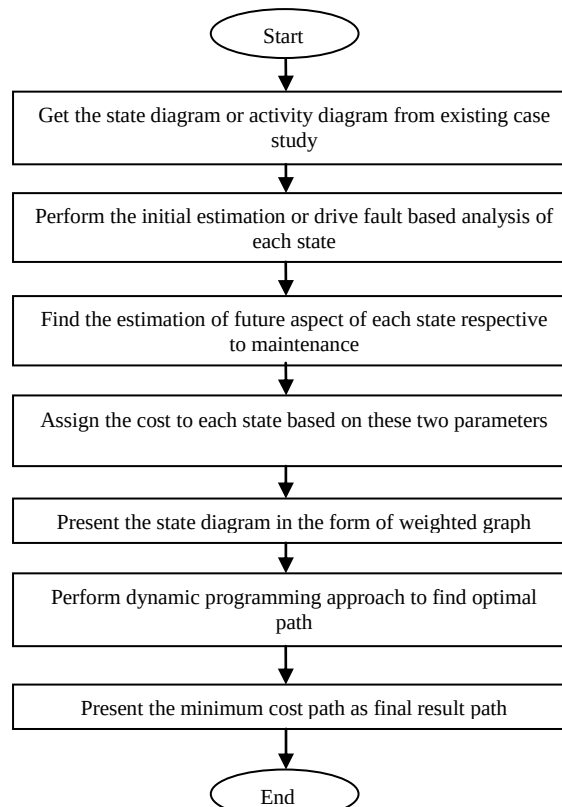


Figure 1 classification of WBT [5].

II. Implementation

Implementation stage involves careful planning, investigation of the existing system and it's constraints on implementation, designing of methods and evaluation of methods. Implementation is the process of converting a new system design into operation. The basic steps the will be included in this work.



Implementation of proposed work is carried out in j2se. On the click of OK button Fig 2 represents a graph where each node consists of a code block having test case append to it. Vertical arrangement shows control flow from top to bottom. Fig 3 shows the output consisting of minimum cost at each level along with the node and minimum total cost in the end.

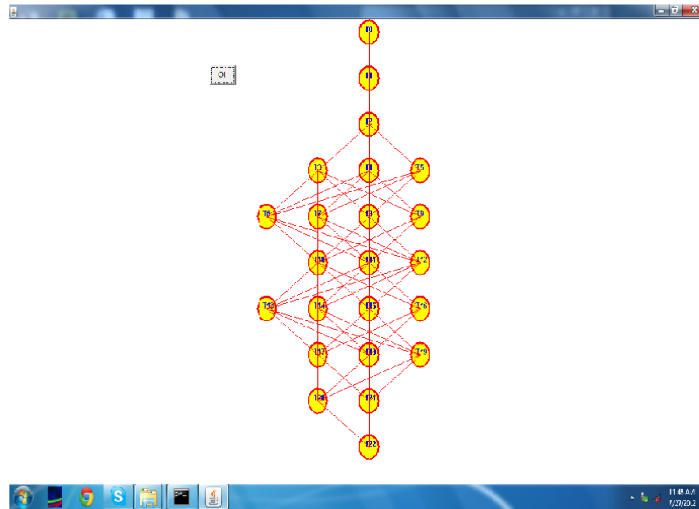


Figure 2 graphical representation of software module.

```

C:\Users\Manisha\Desktop\Java\jdk1.6.0\bin>java work
Stage 8Node 20 30
Stage 8Node 21 16
Min cost Node 21 16
Stage 7Node 17 6
Stage 7Node 18 1
Stage 7Node 19 1
Min cost Node 18 1
Stage 6Node 13 6
Stage 6Node 14 3
Stage 6Node 15 3
Stage 6Node 16 3
Min cost Node 14 2
Stage 5Node 10 6
Stage 5Node 11 1
Stage 5Node 12 1
Min cost Node 11 1
Stage 4Node 6 13
Stage 4Node 7 3
Stage 4Node 8 3
Stage 4Node 9 3
Min cost Node 7 2
Stage 3Node 3 3
Stage 3Node 4 4
Stage 3Node 5 4
Min cost Node 4 4
Stage 2Node 2 3
Min cost Node 2 3
Stage 1Node 1 10
Min cost Node 1 10
Stage 0Node 0 10
Min cost Node 0 10
Minimum Cost Test Case Path 49

```

Figure 3 Calculation of minimum cost path.

III. Conclusion and Future Work

The present work has improved the existing path testing approach by implementing the dynamic programming approach. A software project is the sequence of correlated code modules where some test cases are associated with each module. In this present work test cases are defined respective to the fault occurrence parameter and priority value is assigned to each test case according to future aspects. After this the dynamic programming approach is implemented to find the best path respective to the low cost and less chances of fault occurrence during the testing process.

In this present work dynamic programming approach is used to optimize the path testing. The main parameter taken for the research work is chances of fault occurrence. The work can be extended in different directions

- By using some other optimization approaches such as Genetics or the Neural Network.

- Can also use some different parameter to estimate the test path cost such as number of connections with next and previous code modules etc.

IV. References

- [1] Mandeep, "Introduction and Importance of Software Testing in SDLC", available at <http://www.onestopsoftwaretesting.com/introduction-and-importance-of-software-testing-in-sdlc/#more-338>.
- [2] Dynamic programming at http://en.wikipedia.org/wiki/Dynamic_programming
- [3] Dandoti,V., "White Box Testing: An Overview".
- [4] Glenford J.Myers, "The Art of Software Testing" second edition.
- [5] <http://www.csbd.edu.in/econtent/Software%20Quality%20and%20Testing/UNIT%20II.pdf>