International Journal of Scientific & Engineering Research, Volume 4, Issue 7, July-2013 976
ISSN 2229-5518

# Asymptotic Analysis System for Running Time Recursive Equation

Abhinav Yadav, Dr. Sanjeev Bansal

**Abstract :** The running time of Recursive Algorithms can be described by means of Recursive Equation i.e. Recurrences. Running Time Complexity of such Algorithms can be computed by these recursive equations. The paper intends to introduce a software system that auto generates recurrence equations of form T(n) = a(T(n/b)) + f(n) and creating computational method to solve them and generate running times. At the same time the system also provides for evaluating the submitted solutions and feedback for the solutions. As for now system provides for utilizing the Substitution Method, Master Method for computing running time of the algorithms. Problems related to Order of Growth of functions has also been incorporated. The system contributes toward the learning of algorithm analysis.

**Keywords :** Algorithm, Master Method, Substitution Method, Running Time, Complexity, Learning.

## 1 Introduction

This section discusses the very need of systems that helps in algorithm learning and how they could be of help in learning algorithm theory. The system provides for learning and automatic assessment.

### 1.1 Need of Algorithm Learning System

Learning and Automatic Assessment tools are becoming indispensable part of institutional education. Benefits could be drawn from these tools especially in teaching of data structures and algorithms which are the core and fundamental foundations of computer science. Visualization of manipulation of data structures and algorithms analysis has been undergoing research in the universities around the world. The Innovation and Technology in Computer Science Education group which is sponsored by ACM indicates that such learning and automatic assessment tools are of great help in understanding of core concepts.

### 1.2 Related Work

There exist systems for learning and automatic assessment for teaching data structures through visualization techniques, which have been developed by universities for academic purposes. One such system that has contributed effectively to the field of computer science learning is TRAKALA2 – Software Visualization Group from the Department of Computer Science and Engineering, Helsinki University of Technology. It's an environment for learning data structures and algorithms. The system provides for simulation exercises that can be automatically graded based on the comparisons between the learner made sequences and system's algorithm made sequences.

- *Abhinav Yadav is currently pursuing masters degree program in Computer Science & Engineering, Amity University, India.*
- *E-mail: abhinavanihba@gmail.com*
- *Prof. (Dr.) Sanjeev Bansal currently holds position of Director –Amity Business School, Amity University, India.*
- *E-mail: sbansal1@amity.edu*

### 1.3 Problems Addressed

Most of the existing learning systems for algorithm analysis rely on the concept of visualization of algorithms on data structures. Asymptotic Analysis and Recurrences is a fairly untouched area for creating systems to aid its learning. The key challenge for development of such a system lies in providing a clear problem statement, effectively displaying its solution, and providing interface for automatic assessment. The need is for simulating Substitution Method and the Master method by means of a computation engine for solving recurrence relations.

## 2 Theory

The system that is the subject of the paper intends to provide simulation exercises with the objective of creating deep understanding of recurrence relations and their asymptotic analysis. Substitution Method and Master Method are used to determine asymptotic time complexity of recurrences in the system. A brief description of these methods if provided below.

### 2.1 Substitution Method

The substitution method for solving recurrences entails two steps :
1. Guess the form of the solution.
2. Use Mathematical Induction to find the constants and show that the solution works.
As an example, let us determine the complexity of the following recurrence,

$$T(n) = 2T(n/2) + n \qquad (1)$$

We guess the solution O(nlgn). Our method is to prove that $T(n) \le cn\lg n$ for an appropriate choice of the constant $c > 0$. We start by assuming that this bound holds for n/2, i.e. , that $T(n/2) \le c(n/2)\lg(n/2)$. Substituting into the recurrence yields

$$\begin{aligned} T(n) &\le 2(c(n/2)\lg(n/2) + n \\ &\le cn\lg(n/2) + n \\ &= cn\lg n - cn\lg 2 + n \end{aligned}$$

$= cnlgn - cn + n$
$\leq cnlgn,$
Where the last step holds as long as $c \geq 1$.

2.2 Master Method

The Master Method provides a methodical way for solving recurrences of the form $T(n) = aT(n/b) + f(n)$ where $a>1$ and $b>1$ are constants and $f(n)$ is an asymptotical function. The theorem has three cases in which may recurrences fall in. The three cases of the
Master Theorem are:

1. If $f(n) = O(nlogba - e)$ for some constant $e > 0$, then $T(n) = \Theta(nlogba)$.
2. If $f(n) = (nlogba)$, then $T(n) = \Theta$ **(nlogba lg n)**.
3. If $f(n) = (nlogba + e)$ for some constant $e > 0$, and if $f(n/b) <= cf(n)$ for some constant $c < 1$and all sufficiently large n, then $T(n) = \Theta$ **(f(n))**.

## 3 SYSTEM DESCRIPTION

The main GUI panel consists of various subpanels which are described subsequently :-

In the substitution method panel numerous recurrences can be generated using the "New Problem" button and their solutions can be viewed using "Solution" button.



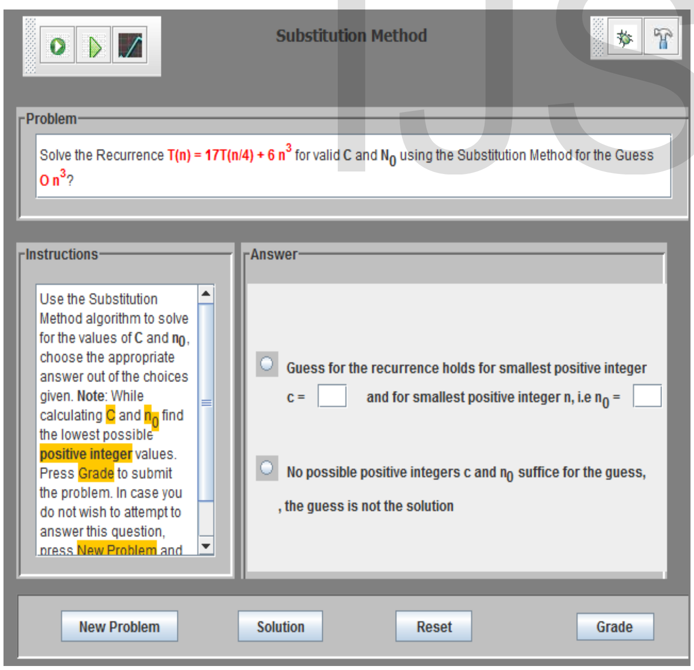


Figure 1. Substitution Method Panel

In this way different kind of recurrences can be viewed and thus this increases the understanding of how the substitution method works and also to how to compute the asymptotic bounds. Alternatively the learner can evaluate his or her solution using the solution generated by the system. Clicking on the solution button invokes the computation engine to generate the solution for the problem statement. The step by step solution is generated and displayed on the solution panel as shown in Figure 2.

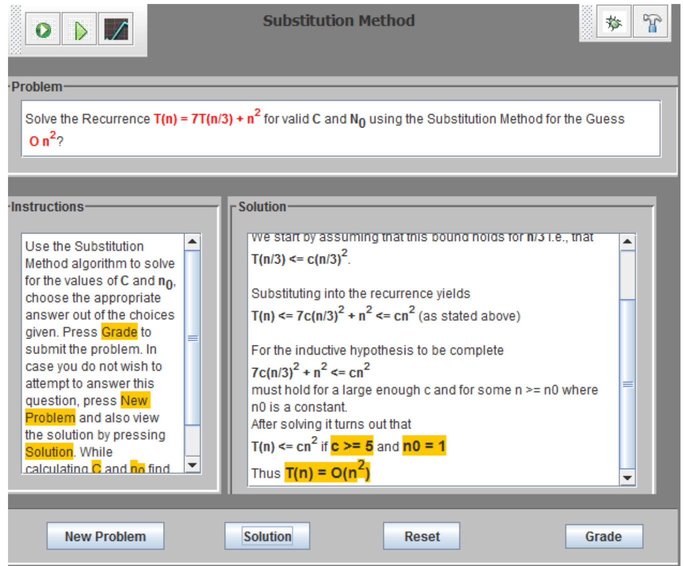


Figure 2. Substitution Method Solution Panel

The system takes the answers given in the answering section and evaluates it with the answers generated by the computation engine. Grade Panel is shown in Figure 3.

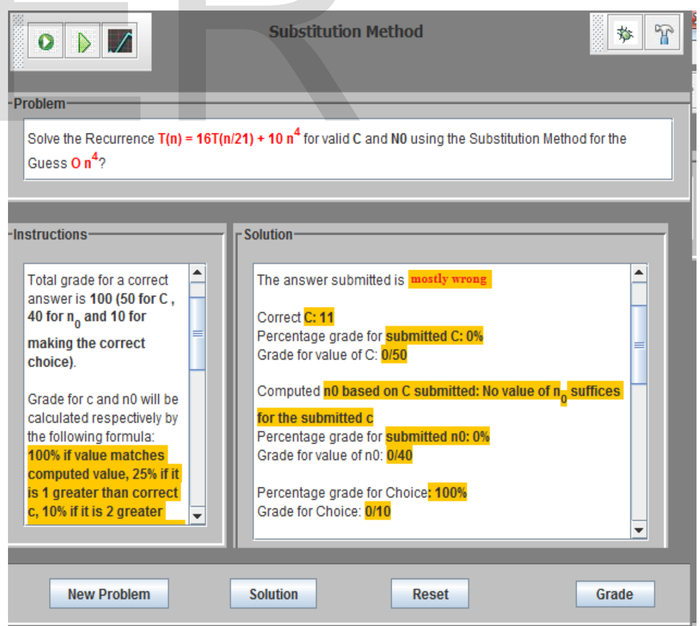


Figure 3. Substitution Method Grade Panel

The system also provides a test panel wherein the user can create custom recurrence equations and test them against custom created bounds. This panel is useful to test the same recurrence against changing upper or lower bounds to understand how c and n0 vary with changing bounds. This panel is shown in Figure 4.

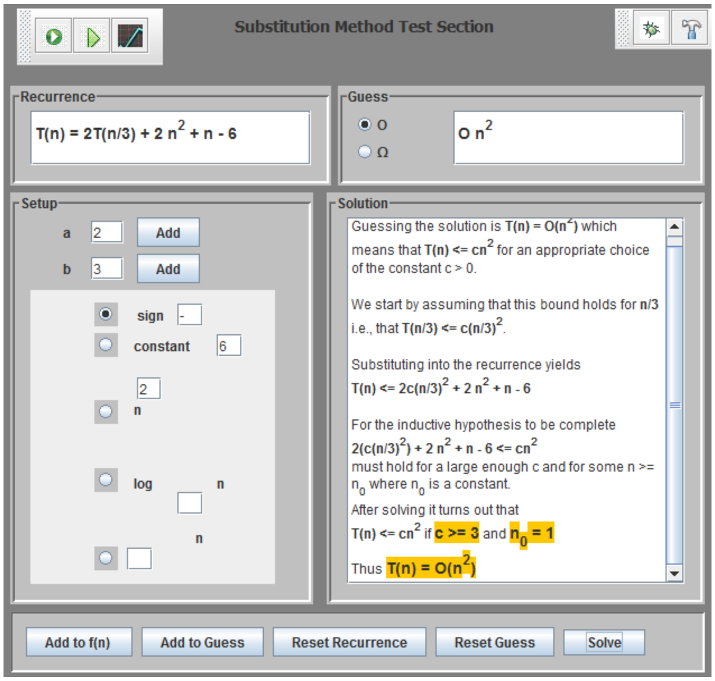


Figure 4. Substitution Method Test Panel

Same as we have discussed so far for Substitution Method there exist module for Master Method. The Master Theorem Panel gives a practice area to solve simulated exercises generated to impart the learning of the Master Theorem. The module generates Recurrence relationship problems and similar to the Substitution Method Panel there are two approaches to learning from them. The learner can either generate numerous problems and view their auto generated solution or choose to test their understanding of the topic by submitting their answers and letting the system evaluate them.

Master Theorem Main Panel is shown in Figure 5.

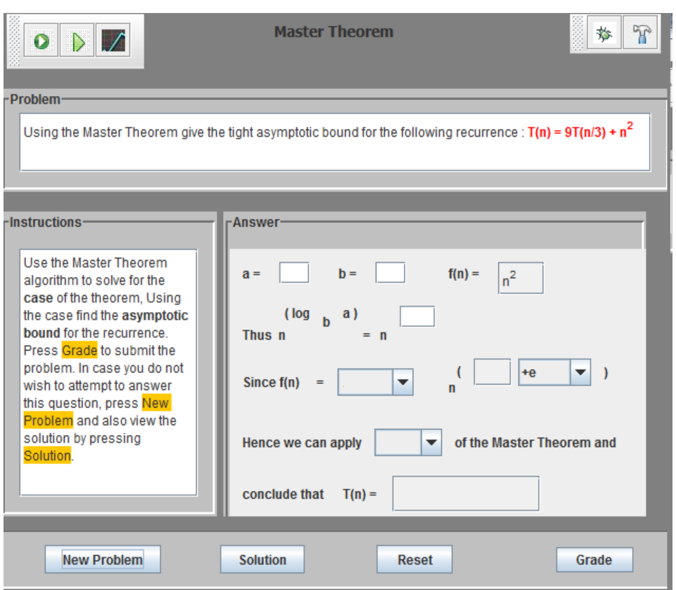


Figure 6. Master Theorem Main Panel

In order to answer the problem, panel provides certain input and combo boxes for the user to input. The user is made to select a series of choices as well as input certain numeric values as his answer. His choices and input values are then converted internally to the asymptotic bound that the user is suggesting through their choices. After entering these values, the combo boxes present options to choose from. According to the choices made the complexity to be submitted it computed dynamically and is updated in the 'T(n) =' box. A completed answer section is shown in Figure 7.

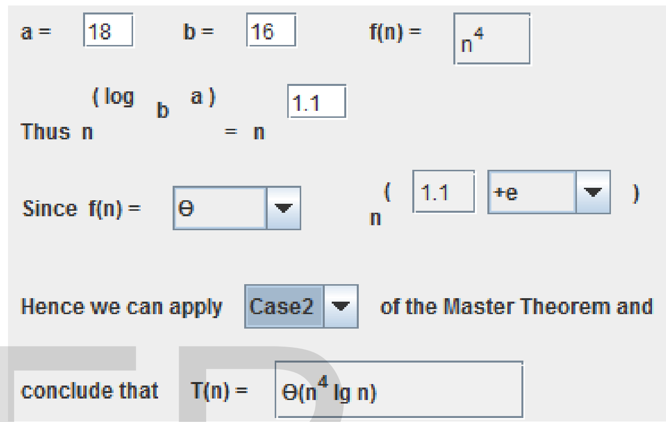


Figure 7. Master Theorem Answer Section Completed

In addition to above discussed Substitution and Master Panel, the system also provides for Order of Growth Panel which helps in learning relative order of growth of various functions.

Order of Growth panel is show in Figure 8.

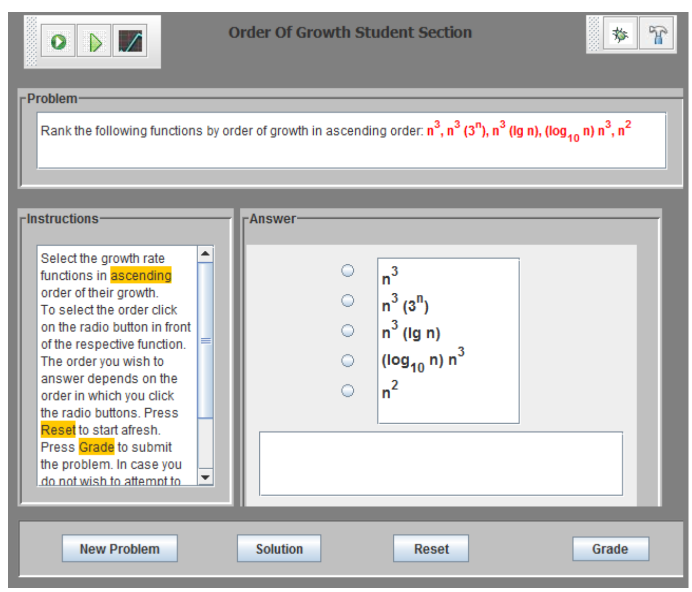


Figure 8. Order of Growth Panel

In order to answer the problem the Answer panel provides the functions with radio button in front of them to select. The order of functions intended to be submitted is order in which the radio buttons are selected. The selected order is above the submit button. The ranking process can be restarted by clicking the reset button.

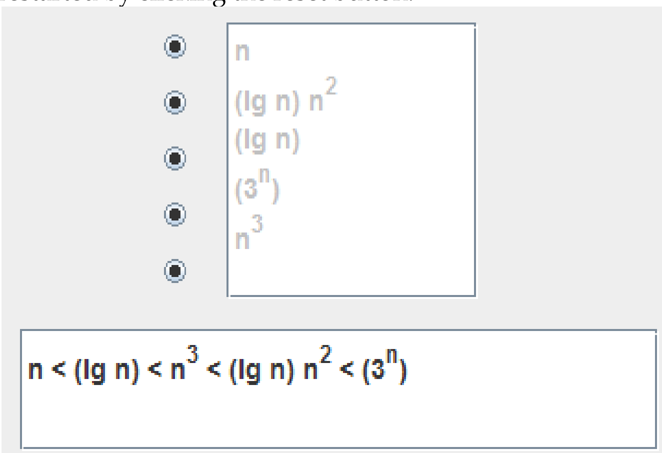


Figure 9. Order of Growth Panel

Clicking on the Solution button invokes the computation engine to generate the solution for the problem statement. The functions are sorted in ascending order of growth and displayed on the solution panel. The system takes the answers given in the answering section and evaluates it with the answers generated by the computation engine.

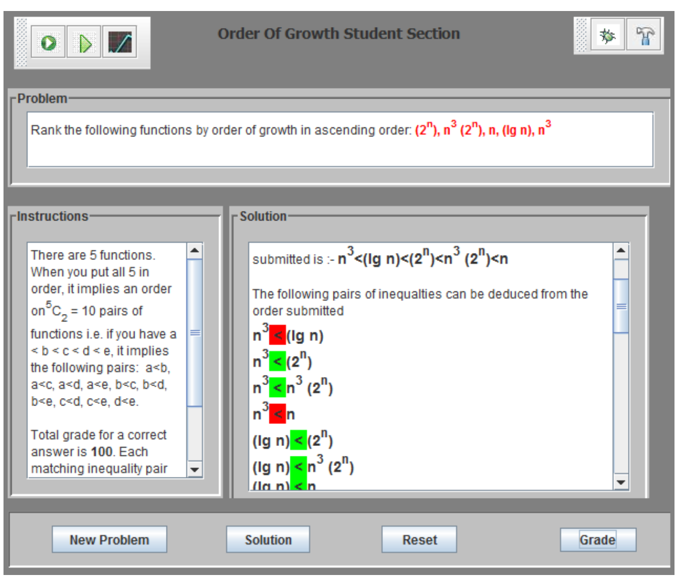


Figure 10. Order of Growth Grade Panel

## 4 Conclusion

In this paper we provide for implementing a System for Asymptotic Analysis of Recursive Equations which caters to the learning f Recurrence Relations and a better and deep understanding of asymptotic time complexities which helps in imparting strong mathematical foundation to Computer Science students. The ability of system to generate worked examples with instant feed fosters a rich learning environment for students. The design of the object model used for the implementation can be reused for creating further enhanced recurrence solving tools and research on the analysis of recurrences.

## 5 Future Work

It is strongly demanded that various extensions to this system should be thoroughly researched and therefore provide an augmentation to the system. We intend to augment this system for Sorting Algorithms as they constitute the fundamental researches in Computer Science. Moreover, these sorting algorithms such as Quick Sort, Merge Sort and Heap Sort are of very importance in Information Technology and Computer Science industry and therefore a clear understanding to these algorithms is of utmost importance for a Computer Science student. We would like to see towards formulating a clear problem statement and thus be incorporated in such systems.

## References

[1] Introduction to Algorithms. Thomas H. Cormen, Charles E. Leiserson, Ronald L.Rivest, Clifford Stein. The MIT Press. 2009.
[2] Algorithm Design. Jon Kleinberg, Eva Tardos. Pearson Education. 2005
[3] Taxonomy of Visual Algorithm Simulation Exercises, Ari Korhonen and Lauri Malmi.2004.
[4] Exploring the Role of Visualization and Engagement in Computer Science Education,Thomas L. Naps, Rudolf Fleischer, Myles McNally. 2003.
[5] Design Pattern for Algorithm Animation and Simulation, Ari Korhonen, Lauri Malmi and Riku Saikkone. 2001.
[6] MatrixPro - A Tool for Demonstrating Data Structures and Algorithms ExTempore, Ville Karavirta, Ari Korhonen, Lauri Malmi and Kimmo St˚alnacke. 2004.
[7] Automatic Assessment of Exercises for Algorithms and Data Structures, Mikko-Jussi Laakso and Tapio Salakoski, 2004.