Assignment 5 - Finding accurate Value Of $\log\left(\frac{x}{x-1}\right)$ Using Series Expansion

Parag Parihar
Roll No:- IIT2016095
iit2016095@iiita.ac.in

Rakshit Sai
Roll No:- IIT2016126
iit2016126@iiita.ac.in

Adarsh Agrawal
Roll No:- IIT2016516
iit2016516@iiita.ac.in

Nilotpal Pramanik
Roll No:- IRM2016501
irm2016501@iiita.ac.in

***Abstract*—Write an efficient algorithm to calculate the value of $\log\left(\frac{x}{x-1}\right)$ for some given input value of x by repeatedly exploring and adding the successive terms in the expansion of the series $\log\left(\frac{x}{x-1}\right)$ = $\frac{1}{x}$+$\frac{1}{2x^2}$+$\frac{1}{3x^3}$ + ...Termination condition is $|S_i$ - $S_{\text{i-1}}| \leq \epsilon$, where S_x denotes value of the series computed up to the x^{th} term and ϵ is an arbitrarily small positive number also taken as an user input for your algorithm. Plot time complexity vs. ϵ Also plot no. of terms considered (summed) vs. ϵ accuracy of the answer produced by your algorithm.**

I. Introduction and Literature Survey

We have to write an efficient algorithm to calculate the value of $\log\left(\frac{x}{x-1}\right)$ for a given input value of x by repeatedly exploring and adding the successive terms in the expansion of the series $\log\left(\frac{x}{x-1}\right)$ = $\frac{1}{x}$+$\frac{1}{2x^2}$+$\frac{1}{3x^3}$ + ... The termination condition is $|S_i$ - $S_{\text{i-1}}| \leq \epsilon$ where S_x denotes value of the series computed up to the x^{th} term and ϵ is an arbitrarily small positive number also taken as an user input for your algorithm. We need to plot time complexity vs ϵ and also need to plot no. of terms considered (summed) vs. accuracy of the answer produced by the algorithm. Along with that, We need to make sure that our algorithm is the best algorithm and the given problem cannot be solved in a much better way than the way we have solved it. We also need to compare it with the previous group which has done the same problem. We need to solve the problem in a much better way than the previous group or we need to prove that it cannot be done in a better way than they have done. The report contains :-
2. Algorithm Design
3. Algorithm Analysis
4. Experimental Analysis
5. Comparison
6. Conclusion

II. Algorithm Design

To construct the algorithm for this particular problem firstly we have to scan the value of **x** as well as **e**,i.e basically the value of ϵ.And pass these two elements as the attributes of the function named **Function** and assigned the return value at **ans**.Then we have assigned the value of $\log\left(\frac{x}{x-1}\right)$ to **original**.As well as the value of **original - ans** has been assigned to **error**.We are calculating the percentage of the error through the formula $\left(\frac{\textbf{original - ans}}{\textbf{original}}\right) * 100$ and storing it into the variable **percentage_error**.

In the function **Function** we are taking the values of **x** and **e** as the attributes and there we are initializing the value of the variables **last** to **0.0**, **latest** to the value of $\mathbf{\frac{1}{x}}$ and **product** to **x** accordingly. After that we are going in to the while loop by satisfying the condition **latest - last** $> \epsilon$.There we are updating the value of **product*x** to **product** as we have to calculate power of **x** for each iteration , the value at **latest** to **last** and the value of **latest** + $\frac{1}{\textbf{product*i}}$ to **latest**.We are coming out of the loop by incrementing the value of **i** which has been already declared as **2** outside the loop.And after all execution the function **Function** is returning the value of **latest** which is value of series expansion to the variable **ans** declared in the **Main** function.

Algorithm 1 Function

```
INPUT: x, $\epsilon$
OUTPUT: value of series expansion of $\log\left(\frac{x}{x-1}\right)$ according to $\epsilon$
$last \leftarrow 0.0$
$latest \leftarrow \frac{1}{x}$
$product \leftarrow x$
$i \leftarrow 2$
while latest - last $> \epsilon$ do
    $product \leftarrow product * x$
    $last \leftarrow latest$
    $latest \leftarrow latest + \frac{1}{product*i}$
    $i \leftarrow i + 1$
end while
return latest
```

Algorithm 2 Main Function

```
Scan the value of x where x>1 and $\epsilon$
$ans \leftarrow$ function(x,$\epsilon$)
$original \leftarrow$ Exact value of series
$error \leftarrow$ original - ans
$percentage\_error \leftarrow \left(\frac{\textbf{original - ans}}{\textbf{original}}\right) * 100$
```

III. Analysis and Discussion

In this section we have analysed all the cases including time complexity and graphs and notations.

A. *Time-Complexity——-*

*1) **Best-Case**:* The best case for our algorithm consists no iteration of while loop, and this will happen when ϵ is greater than equal to $\frac{1}{x}$. In this case no. of instructions are 8 .However this will not give accurate value of series expansion for the given x.

*2) **Worst-Case**:* The worst case for a given x would be when ϵ is just greater than zero i.e when ϵ is positive and has the least value possible. This is considered to be the worst case because the algorithm will have to calculate values for maximum number of terms.
The time complexity depends on ϵ as well as the x given. The bigger the x ,the more number of terms is required to make the required difference less than ϵ. Therefore the worst case is when ϵ is positive and as small as possible and given x is as large as possible. The given problem statement asks us to plot time units versus ϵ for any fixed value of x.

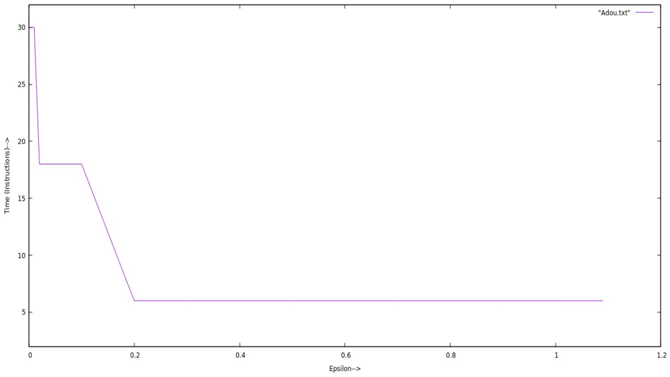

Fig.1 Number of instructions vs ϵ

The question asks us to plot the graph between the number of terms considered and the accuracy percentage. Therefore plotting the number of terms considered and the percentage error would be equivalent.We can see that in the graph as we are increasing no. of terms the error is decreasing drastically. For the following graph we have taken the value of ϵ 0.

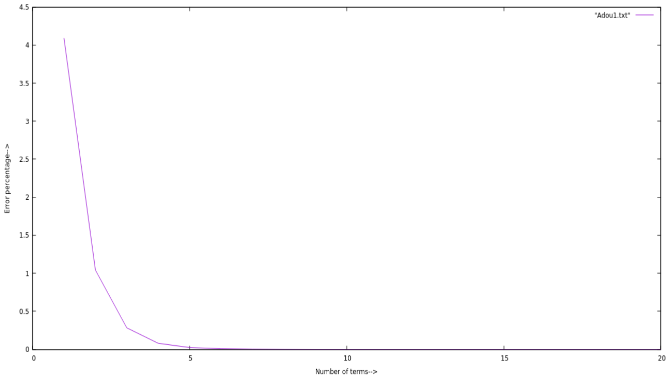

Fig.2 : No of terms considered vs percentage error

IV. Experimental Analysis

We revised the commands and basic functions of **GNUPlot** to plot graphs and other relevant analysis related to our algorithm. While making this report we also learnt the basics of making reports using **LATEX** in IEEE format using **IEEEtran** class.

For first graph that is between ϵ and time , we have taken any random value of x and for different value of ϵ we have calculated value of time using count in our code. In this way we got following table for ϵ and time(Instructions) :-

ϵ	**Instructions**
1.02000000	6
1.01000000	6
0.90000000	6
0.70000000	6
0.50000000	6
0.30000000	6
0.20000000	6
0.09800000	17
0.07000000	17
0.05060000	17
0.03000000	17
0.02000000	17
0.01000000	28
0.00900000	28
0.00700000	28
0.00500000	28
0.00300000	28

From above table, we can see that as the value of ϵ is increasing the no. of iterations are increasing but after certain value of ϵ no. of instruction gets constant value.

For second graph that is between No. of terms considered vs Percentage Error, we have taken any random value of x and for different no. of iterations we have calculated percentage error In this way we have got following table :-

No. of Terms	**Error**
10	0.0000559
9	0.00018249
8	0.00060016
7	0.00199239
6	0.00669119
5	0.02280134
4	0.07918688
3	0.28217482
2	1.04337959
1	4.08819868

V. COMPARISON

By comparing the algorithm which we presented with the algorithm designed by the previous group we can tell that the one which we designed is much better and optimized since the complexity of our algorithm is $O(n)$ where **n is no. of iterations (for while loop)**. and that of the other group is $O(n + log(n!))$. The reason the other group have arrived with the time complexity of $O(n + log(n!))$ is that they have used the **power function** in order to calculate the power of a number which is taking them **log(i!)** where i is the current power of x, for each iteration and the power function is in the while loop which is making the loop run until the condition in the loop is satisfied.

As we can understand in our algorithm in each iteration of while loop there is constant no. of instructions (say c) , so our algorithms complexity is $O(cn)$ or $O(n)$, but in other groups algorithm, instructions for while loop is like $c+log(2)+c+log(3)+c+log(4)+c+log(5)+....+c+log(n)$, i.e, which is equivalent to $O(nc+log(n!))$ or $O(n+log(n!))$ where **c is constant instruction in each iteration**.

VI. Conclusion

In the assignment we have designed an algorithm to find the value of $\log\left(\frac{x}{x-1}\right)$ using the **Taylors Series Expansion**. We had to give a value of x and the result will be $\frac{1}{x}$ + $\frac{1}{2x^2}$ + $\frac{1}{3x^3}$ + ... The expansion length also depends on the value of x. The series must follow the inequality $|S_i$ - $S_{\text{i-1}}| \leq \epsilon$. Here S_x is the value of the series computed till the xth term and the term ϵ is very very small and is chosen randomly. Epsilon is taken as the input in the algorithm.

We also compared our algorithm with the one which is submitted by the previous group. We have optimized our algorithm even more and the complexity of our algorithm is O(n) where **n is no. of iterations** .We have also plotted the graph for **Complexity vs** ϵ and **Number of terms considered vs the accuracy of the answer produced by the algorithm**.

References

[1] Introduction To Algorithms (Second Edition) by Thomas H.Cormen, Charles E.Leiserson , R.L.Rivest and Clifford Stein.

[2] The C Programming Language by Brian Kernighan and Dennis Ritchie.

[3] https://www.geeksforgeeks.org/

[4] https://www.stackoverflow.com/