Implementing and Analyzing Custom Square Root Funciton

Yousha Munowar

McMaster University

MECHTRON 2MP3

Pedram Pasandide

September 26th 2024

Method

The Newton-Raphson method is a well-known iterative method to approximate the roots of a function. This method was implemented using the following process. Newtons equation for square root can be used, in which the more iterations the equation goes through, the more accurate of a result. Knowing the Newtons-Raphson Method for Square Root is $x_{n+1} = \frac{1}{2}\left(x_n + \frac{a}{x_n}\right)$, we implement this method into the code and solve for the square root without the pre-built function within the math library. This can be done through creating three functions, one to solve for absolute value, another to multiply with respect to power of 10, and the sqrtUser function to execute all of them. The first two functions will be solely for precision and limiting the program to know when to stop and keep it efficient.

Time Complexity

Time complexity is a core concept in the software world as efficiency is one of the key skills every expert programmer must have. The Time complexity of Sqrt () in the math library of the programming language C is O (1), thus constant time complexity. This is the most efficient time complexity, and with C being one of the fastest programming languages, this process occurs extremely quickly. However, with the sqrtUser () function which utilizes Newtons-Raphson Method for Square Root has a time complexity of O (log n), thus a logarithmic time complexity. Thus, the Sqrt () method is much faster. This is due to the built-n sqrt () function leverages optimization methods which are designed for a constant time complexity, thus regardless of the size of the input, it will remain constant. However, with sqrtUser(), it involves various steps and processes to attain the result, as it is input size dependent, and the larger an input size or number of decimals, the longer the program will take to achieve the result. In summary, although the custom method with time complexity Olog(n) is without a doubt faster than majority of time complexities, it is no match for the built in hardware-optimized function in the lnagugae C.

Compiling and Running

To compile and run this program, the following can be done in the terminal:

1. Cd ASSIGNMENT (open directory)
2. Gcc sqrtuser.c (compile the code file)
3. ./a.out (execute)

This should successfully open the respective directory, compile the file for the machine to understand, and execute it.

Appendix

```
#include <stdio.h> //include C language standard library

double absolute(double num) { //create a function to return the inputs absoluete value
if (num < 0) {
return (num * -1); //if the number if negative, multiply it by -1
} else {
return num; //if it is not negative, return the number as in
}
}

double PowerTen(int num){ //create a function to calculate the power of 10's
double result = 1.0; //initlaize the vairbale as 1
if (num >= 0) //if the input is more than 0,
{
for (int i = 0; i < num; i++) //implement a for loop for repeated action
{
result *= 10; //multiply the number by 10 the same number of times as the input dictates
}
}
else{
for (int i = 0; i < -num; i++) //if the input is a negative, implement for loop for repeated action
{
result /= 10; //divide the number by 10 the same number of times as the input dictates
}
}
return result; //return the result of the loop
}
```

```
double sqrtUser(double number, int n) { //main funcion that will take the inputs
double difference = PowerTen(-n); //this will call the PowerTen function to calculate the precision
double Current_guess = number / 2; //an initial guess for the newtons method
double previous_guess; //initlaize the vairable of previous guess

do {
previous_guess = Current_guess; //the prevoius guess now becomes the current guess
Current_guess = 0.5 * (Current_guess + number / Current_guess); //do the newtons method equation
} while (absolute(Current_guess - previous_guess) > difference); //keep going untill the aboslute
//difference of current guess and previous guess is more than the differnece

double factor = PowerTen(n); //find the power of ten of the deicaml place input
Current_guess = (int)(Current_guess * factor) / factor; // multiply the current guess by factor, and
//then turn it into an integer, earsing any decimals, and then divide by factor to return the decimal place

return Current_guess; //return the final result
}

int main() { //start the program
double num; //inital vairable num as the number you want to root
int num_decimals; //inital the vaiable num_deciamls as the number of decimals yoou want as an integer

printf("Enter the number you would like the square root of: ");//Print a statment to ask for the number to square root
scanf("%lf", &num); //take the input from user, scan it and store it in the vairable num
printf("Enter the number of deciamals you would like: ");//Print a statment to ask for the number of deicmals
scanf("%d", &num_decimals);//take the input from user, scan it and store it in the vairable num_decimals

double result = sqrtUser(num, num_decimals); //execute the square root function using the user inputs

if (result > 0) {
printf("The square root of %.15f to %d decimals places is %.15f" ,num,num_decimals ,result);//as long as the result is
more than 0
// print it with respect to 15 deicmals
}

}
```

References

Josh Morrison 7. (1956, June 1). *What is Newton-Raphson Square Method's time complexity* Stack Overflow. https://stackoverflow.com/questions/5005753/what-is-newton-raphson-square-methods-time-complexity

GeeksforGeeks. (2024, June 25). *Sqrt() function in C*. https://www.geeksforgeeks.org/sqrt-function-in-c/

Chatgpt. (2022, November 30). https://chatgpt.com/

For additional information on APA Style formatting, please consult the APA Style Manual, 7th Edition.