Fundamentals of Data Structures

Laboratory Projects

Performance Measurement (POW)

anonymous

Date: 2021-10-03

Chapter 1: Introduction

There are at least two different algorithms that can compute X^N for some positive integer N.

Algorithm 1 is to use N−1 multiplications.

Algorithm 2 works in the following way: if N is even, X^N=X^N/2×X^N/2; and if N is odd, X^N=X^(N−1)/2×X^(N−1)/2×X. Figure 2.11 in our textbook gives the recursive version of this algorithm.

Chapter 2: Algorithm Specification

Use three relatively simple functions to represent the corresponding algorithms. The program solves the task using three similar linear data structures.

Chapter 3: Testing Results

The test results are shown below

```
N       1000      5000      10000      20000      40000      60000      80000      100000
Algorithm 1:
100000     60000     30000     10000     6000     3000     1000     600
315.000000  916.000000  910.000000  602.000000  724.000000  574.000000  252.000000  179.000000
0.315000  0.916000  0.910000  0.602000  0.724000  0.574000  0.252000  0.179000
0.000003  0.000015  0.000030  0.000060  0.000121  0.000191  0.000252  0.000298
Algorithm 2(Recursive):
100000     60000     30000     10000     6000     3000     1000     600
306.000000  1619.000000  1596.000000  1040.000000  1298.000000  587.000000  436.000000  244.000000
0.306000  1.619000  1.596000  1.040000  1.298000  0.587000  0.436000  0.244000
0.000003  0.000027  0.000053  0.000104  0.000216  0.000196  0.000436  0.000407
Algorithm 2(Interative):
100000000     60000000     30000000     10000000     6000000     3000000     1000000     600000
2939.000000  2205.000000  1171.000000  396.000000  264.000000  137.000000  49.000000  29.000000
2.939000  2.205000  1.171000  0.396000  0.264000  0.137000  0.049000  0.029000
0.000000  0.000000  0.000000  0.000000  0.000000  0.000000  0.000000  0.000000
请按任意键继续. . .
```

The four lines represent Iterations(K), Ticks, Total Time and Duration.

Chapter 4: Analysis and Comments

The time complexities of the algorithms are O(N) and space complexities are $O(N^2)$. The output format needs to further optimized.

Appendix: Source Code (in C)

```c
double powNormal(double x, int n) {
    double y = x;
    int i;
    for(i=1; i<n; i++) {
```

```
        y *= x;
    }
    return y;
}

//If N is even, X^N = X^(N/2) × X^(N/2); and if N is odd, X^N = X^((N-1
)/2) × X^((N-1)/2) × X
double powHardInterative(double x, int n) {
    double y = 1;
    while(n > 0) {
        if(n & 1) {
            y = y * x;
        }
        n /= 2;
        x *= x;
    }
}

double powHardRecursive(double x, int  n) {
    if(n == 0) {
        return 1; //When n = 0
    } else if(n == 1) {
        return x;
    } else {
        if(n & 2) {
            return powHardRecursive(x, n/2) * powHardRecursive(x, n/2)
* x;
        } else {
            return powHardRecursive(x, n/2) * powHardRecursive(x, n/2);
        } //Use the recursive way to realize the algorithm 2
    }
}
```

Declaration

I hereby declare that all the work done in this project titled "pr1" is of my independent effort.