CMSC 142 MP1: Towers of Hanoi

Juancho Meneses
Gimel Velasco
jrv.meneses@gmail.com
gfvelasco@up.edu.ph

ABSTRACT

This paper presents an algorithm for solving the Towers of Hanoi puzzle. For analyzing the performance of the algorithm, the behaviour of the time it takes to solve an n-disked Towers of Hanoi puzzle is observed.

1 INTRODUCTION

Towers of Hanoi is a puzzle game which involves three vertical pegs and a set of disks of varying sizes having holes through their centers. Towers of Hanoi, according to most people's belief, was invented in 1883 by a French mathematician named ftdouard Lucas, though his part in its invention was questioned.[1] The main objective of the game is to transfer all the disks from one peg to another in the same order as they were in the first peg which is increasing in size from top to bottom. There are two rules to follow in transferring the disks. The first rule is you may only move one disk at a time. The second rule is you cannot put a bigger disk on top of a smaller disk. These rules made the game it a puzzle. According to legends of some countries in Asia, monks were solving this puzzle with a set of 64 disks. These monks believed that the moment they solved the puzzle which means they have completely moved the disks from peg A to peg B, the world is going to end.[3]

The objective of this paper is to understand further the problem (Towers of Hanoi)and its running time by developing a computer program which solves it and records the time of solving it. The method used in developing and writing the program and the results obtained from the program will be discussed in the next sections.

2 METHODOLOGY

The method used for solving the towers of Hanoi of n disks is recursion.

```
Function towerofhanoi(n, source, dest, spare):

    IF n==1, then:
        move disk 1 from source to dest //base case
    ELSE:
        towerofhanoi(n - 1, source, spare, dest) // Step 1
        move disk n from source to dest // Step 2
        towerofhanoi(n - 1, spare, dest, source) // Step 3
    END IF
```

Figure 1: Pseudo code of Towers of Hanoi

The pseudo code of the program is shown in Figure 1.

```
function towersOfHanoi(N, src, aux, dst)
    if N == 1
        %fprintf('Disk %d from %c to %c.\n', N, src, dst);
    else
        towersOfHanoi(N-1, src, dst, aux);
          %fprintf('Disk %d from %c to %c.\n', N, src, dst);
        towersOfHanoi(N-1, aux, src, dst);
    end
end
```

Figure 2: MATLAB code of Towers of Hanoi

The program was created using MATLAB programming language. The code of the program is shown in Figure 2 and 3.

The program was run in an Asus X453M laptop with Intel Celeron (Quad-Core) processor, 4 GB RAM and 64bit operating system. The program was run from n=3 (n is the number of disks) up to n=35. Three trials were made for each iteration and the running time of each trial was recorded. The average running time of each iteration was also computed and then plotted into graph. Other applications such as Google Chrome, TexMaker, and Bitdefender were also running while the program was being ran.

```
for N=3:35
    fprintf('%d Disk/s\n', N);
    avgTime = 0;

    for trial=1:3
        tic
        fprintf('\tTrial %d: ',trial);
        towersOfHanoi(N,'A','B','C');
        time = toc;
        fprintf('%.6fs\n',time);
        avgTime = time + avgTime;
    end
    avgTime = avgTime/3;
    fprintf('Average Time: %.6fs\n----------\n',avgTime);
end
```

Figure 3: MATLAB code of Towers of Hanoi 2

3 RESULTS AND DISCUSSIONS

The program developed outputted the running time of each of the three trials and the average of them per iteration from n = 3 to n = 35. A sample run of the program with iterations from 3 disks to 6 disks is shown in Figure 4.

```
>> mp1_main
3 Disk/s
    Trial 1: 0.018918s
    Trial 2: 0.001807s
    Trial 3: 0.002006s
Average Time: 0.007577s
--------------------
4 Disk/s
    Trial 1: 0.000694s
    Trial 2: 0.002622s
    Trial 3: 0.000433s
Average Time: 0.001250s
--------------------
5 Disk/s
    Trial 1: 0.000712s
    Trial 2: 0.000239s
    Trial 3: 0.000214s
Average Time: 0.000388s
--------------------
6 Disk/s
    Trial 1: 0.000204s
    Trial 2: 0.000364s
    Trial 3: 0.000155s
Average Time: 0.000241s
--------------------
```

Figure 4: Sample run n = 3 to n = 6

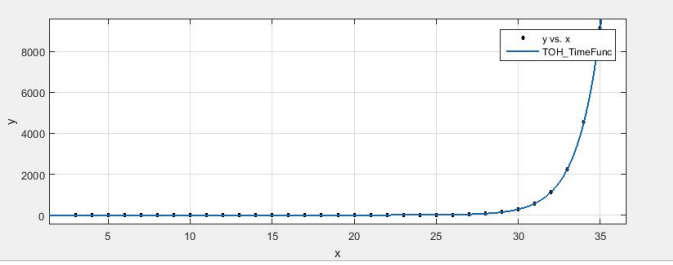

Figure 5: Graph of the tabulated results with curve-fit

The results gathered from the main run of the program which from 3 disks to 35 disks is shown in below and they were plotted into graph with curve-fitting which is shown in Figure 5 where y is the time in seconds and x is the number of disks.

Number of Discs	Runtime (seconds)
n=3	0.002893
n=4	0.001253
n=5	0.000283
n=6	0.000246
n=7	0.000218
n=8	0.000237
n=9	0.000338
n=10	0.000420
n=11	0.000709
n=12	0.001372
n=13	0.002489
n=14	0.004625
n=15	0.009304
n=16	0.017761
n=17	0.035742
n=18	0.074703
n=19	0.142778
n=20	0.279510
n=21	0.558569
n=22	1.117595
n=23	2.229547
n=24	4.455078
n=25	8.869135
n=26	17.748603
n=27	35.532751
n=28	71.179480
n=29	142.166025
n=30	284.485094
n=31	562.532558
n=32	1125.245287
n=33	2258.260900
n=34	4546.780062
n=35	9156.502660

3.1 Extrapolating Results

Since the computer/laptop used cannot finish the iterations from 36 disks up to 10,000 disks, the results of these other iterations were extrapolated. For extrapolating the results, the MATLAB Curve Fitting is used. The Exponential Curve Fitting method is used since it best fits the behaviour of the elapsed time as the number of discs progresses. The general model of the Exponential Curve Fitting with one term is

$$f(x) = a * exp(b * x) \tag{1}$$

. The general model of the Exponential Curve Fitting with two terms is

$$f(x) = a * exp(b * x) + c * exp(d * x) \tag{2}$$

where f(x) is the time function in seconds and x represents the number of discs and the coeffients a, b, c and d are to be solved. The function form with two terms is used since it was fitter than that of the function form with only one term.

After plugging in the results of the runtime from 3 discs to 35 discs, the following coefficients were solved by MATLAB Curve Fitting:

$a = 1.123e - 05$

$b = 0.4842$

$c = 1.572e - 07$
$d = 0.7074$
therefore the yielded disc versus time function of the Towers of Hanoi is where the Root-Mean-Square-Error (RMSE) yielded 0.739. On the other hand, the one-termed function yielded an RMSE of 1.923. (Figure ‘6)

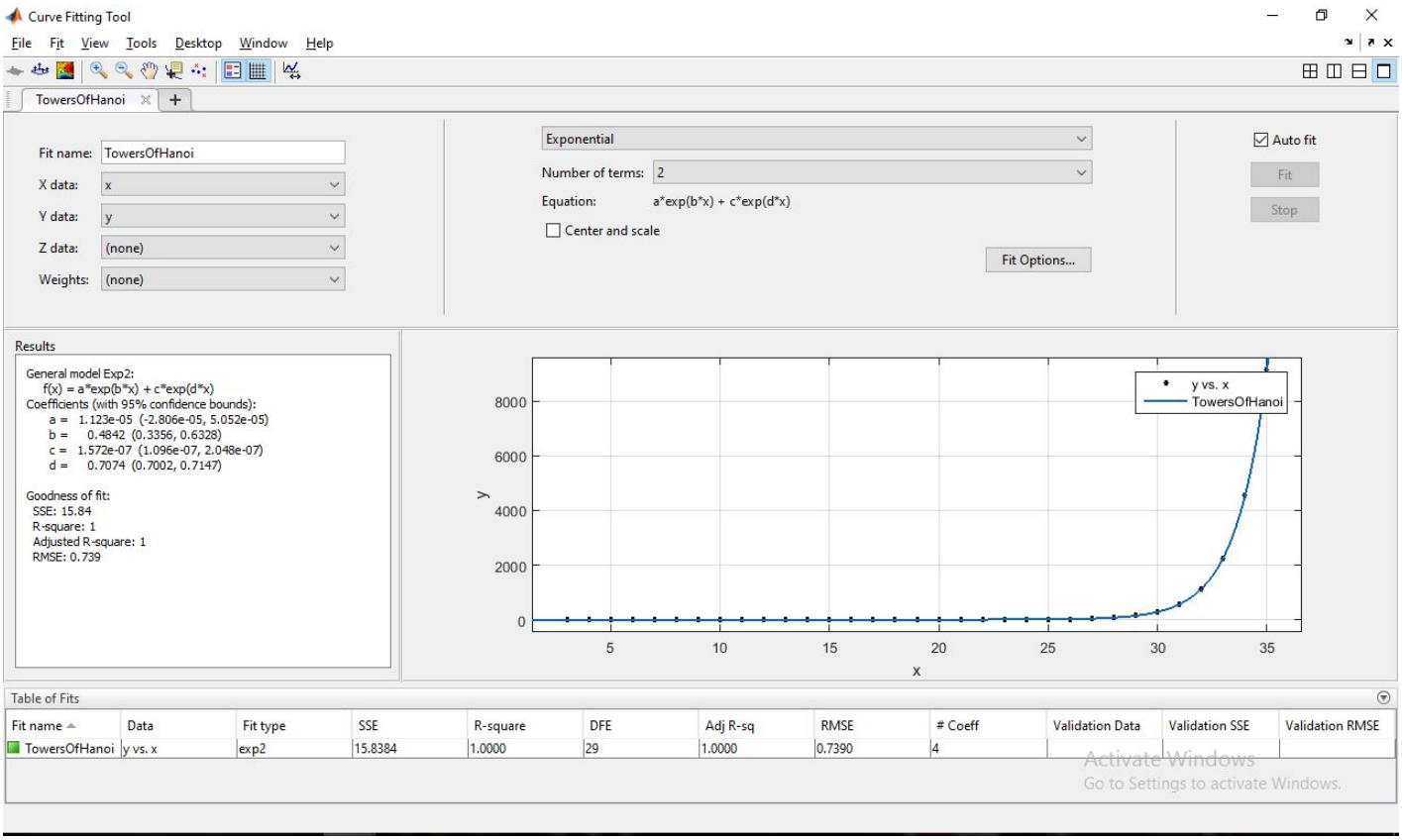

Figure 6: Curve-fitting results

Using the function stated above, the Towers of Hanoi with 64 discs was predicted to have a total runtime of 7.2202e+12 seconds (232 130 years). And the Towers of Hanoi with 10 000 discs yielded a time of infinite value.

4 CONCLUSIONS

A Towers of Hanoi puzzle with 3 to 21 discs was solved by the algorithm under a second. With 22 discs and above, it can be easily observed that the time it takes to solve the puzzle doubles in every additional disc. Based on the results of the tests, the yielded runtime for the algorithm to solve the Towers of Hanoi puzzle showed an exponential behaviour with respect to the increasing number of discs.

REFERENCES

[1] Tower of Hanoi. https://www.britannica.com/topic/Tower-of-Hanoi. (????). Accessed: 2017-02-17.
[2] Tower of Hanoi and Big O. http://www-bcf.usc.edu/~stejada/csci101/slides/2012/TowerHanoi.pdf. (????). Accessed: 2017-02-17.
[3] Thomas Cormen and Devin Balkcom. Towers of Hanoi. https://www.khanacademy.org/computing/computer-science/algorithms/towers-of-hanoi/a/towers-of-hanoi. (????). Accessed: 2017-02-17.