Investigation on the Tower of Hanoi

Je-Yu Leo Chou

April 4th 2021

1 Introduction

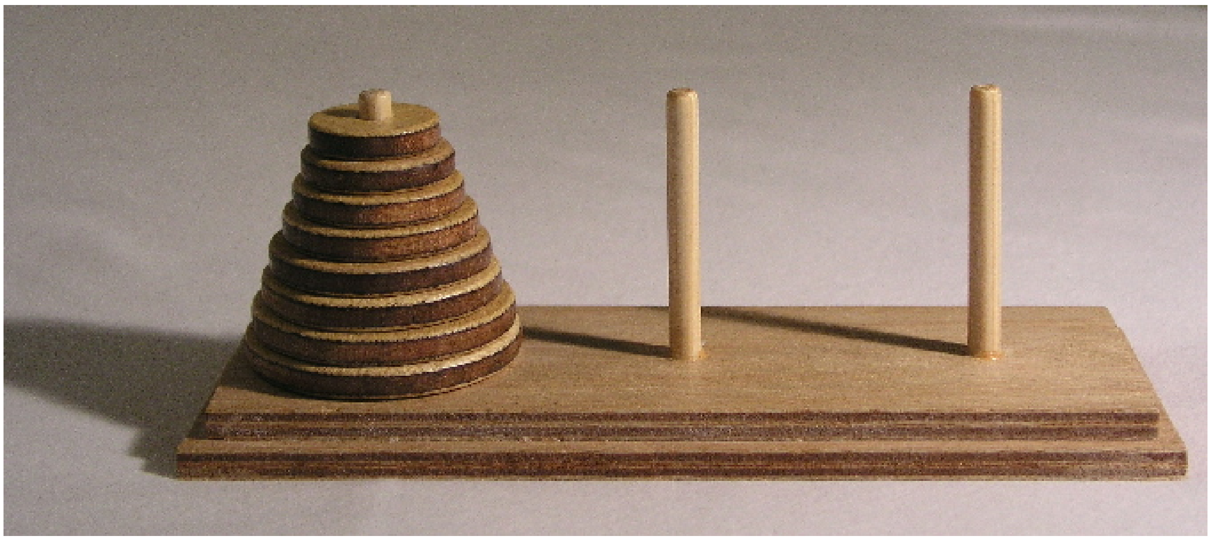


Figure 1: A model of the Tower of Hanoi with 8 disks, and 2 additional, empty rods. Created in 2005 by user Ævar Arnfjörð Bjarmason and shared under the CC BY-SA 3.0 license

1.1 History of the Tower of Hanoi

The tower of Hanoi is a mathematical puzzle invented by a French mathematician named Édouard Lucas in the 19th century [1]. It was said that if one follows the assigned rule for 64 disks, the time to complete the task would mean the end of the world (there is also another legend in whic this is a test for young priest in hindu temples) [1].

1.2 Problem Statement

The tower of Hanoi is a simple, iterative mathematical puzzle involving three diagrams [1]. At the start, there is an arbitary amount of disks placed at the first peg with two additional empty pegs as shown in figure 1.

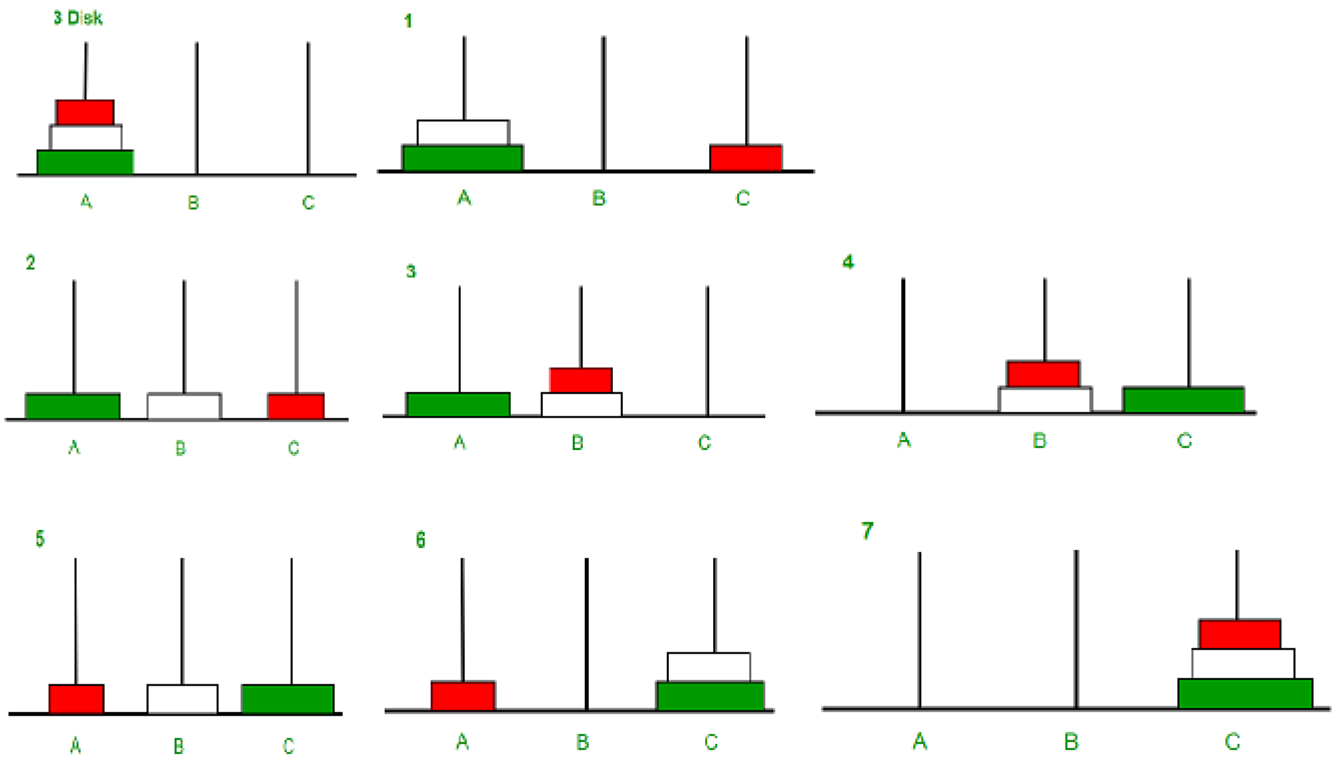


Figure 2: A method to solve the puzzle with 3 disks. Note that this fashion requires the minimal amount of moves of seven moves. This image, and its original article, was on GeeksforGeeks, contribued by Rohit Thapliyal, and shared under the CC BY-SA 3.0 license

.

The problem is simple: Move all disk from the first pole to the third pole [1]. There are, however, two additional rules: (a) you may only move one disk at a time, and (b) the smaller disk must be on top of a larger disk [1]. Figure 2 is the minimal move solution for 3 disks.

1.3 Motivation

The tower of Hanoi is a longstanding problem that is often presented to older kids as well as programmers as the problem itself isn't hard and is an iterative problem (the same set of instruction is used over and over again) [1]. This famous puzzle was presented to me multiple times throughout my schooling, yet I never saw the real world significance of it. However, as a high schooler and a programmer now, I saw the application of it in many types of problems and its importance, as recursion has proven useful in many disciplines, such as creating tries and solving other famous mathematical puzzles such as Knight's Moves Problem.

2 Investigation

2.1 Finding a Pattern

In order to investigate the minimal move (thus the most efficient algorithm) for this problem, it is important to find a patterns in which to solve the problem. To find it, we will run through some cases in order to establish its relation. More specifically, we will be looking at cases where n=1; n=2; n=3, and finally when n=k where n is the number of disks and k $\in \mathbb{N}$. Below, A, B, and C will refer to the three pegs individually, where peg A is the peg with all the disks at the start, peg B is the helper peg, and peg C is the final peg where all disk should go in the end.

2.1.1 n=1

This is the most trivial case, and one that will become important later. To do this, all we need to do is the move disk one from A to C

2.1.2 n=2

In this case, there are a minimal of three steps to take, as listed:

1. Move disk 1 from A to B
2. Move disk 2 from A to C
3. Move disk 1 from B to C

2.1.3 n=3

This is the case where the pattern starts to emerge. We first examine the seven steps required:

1. Move disk 1 from rod A to rod C
2. Move disk 2 from rod A to rod B
3. Move disk 1 from rod C to rod B
4. Move disk 3 from rod A to rod C
5. Move disk 1 from rod B to rod A
6. Move disk 2 from rod B to rod C
7. Move disk 1 from rod A to rod C

These steps are represented visually in Figure 2. From this, we can see a clear pattern where in order to move all three disks, we first have to first n-1 disks to a seperate, auxiliary peg, then move the final disk to the final peg, then move the n-1 disks from the auxiliary peg to the final peg. This brings us to the n=k scenario

2.1.4 n=k

As mentioned before, in order to solve for an arbitrary amount of pegs, there are only four steps to follow:

1. Move $k-1$ pegs to the auxiliary peg
2. Move the k^{th} peg to the final peg
3. Move the $k-1$ pegs from the auxiliary peg to the final peg

This is an iterative problem in mathematics that can be solved with recursion. Recursion refers to solving a problem by first solving a smaller version problem until reaching a trivial point, where the function stops calling on itself. This can be seen from the previous steps [3].

2.2 Finding an Expression

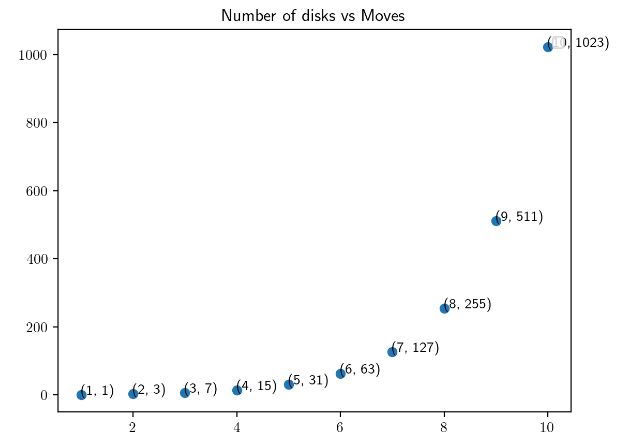
To find the minimal moves required to solve the problem, the following table was created:

Table 1: Comparision on minimum steps with the number of disks

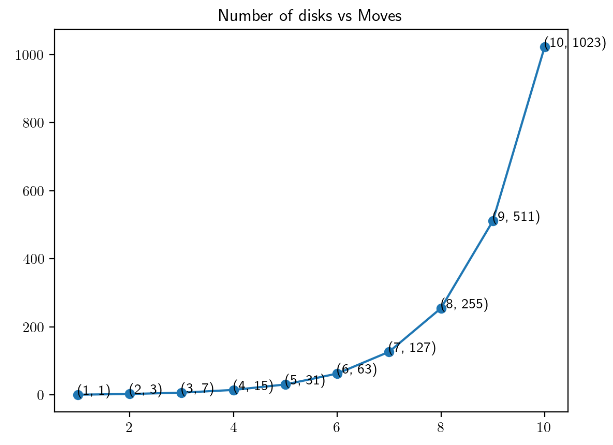
Number of Disks	Minimum Steps	Difference
0	0	N/A
1	1	1
2	3	2
3	7	4
4	15	8

In order to come up with an equation to find out what the equation is for the minimal moves required for n disks, this investigation offers its examination along with its proof.

2.2.1 Graphs

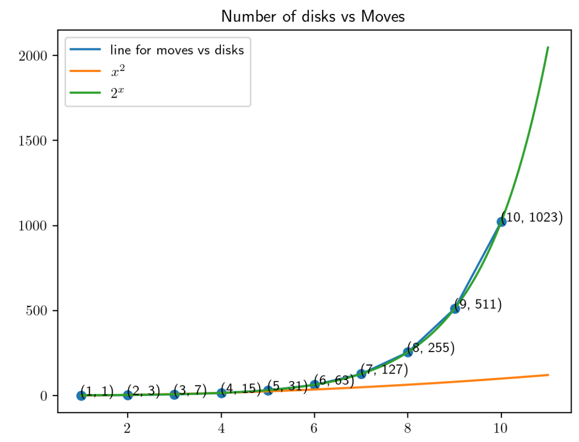


(a) scatter plot of disks vs moves

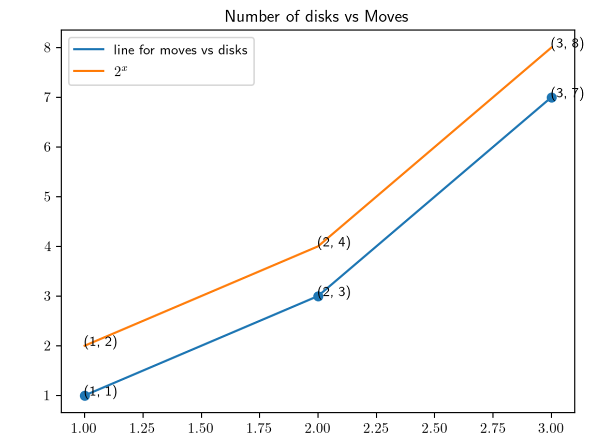


(b) Fit of a line over the scatter plot

Figure 3: Tracing out the graph of disks vs moves through first a scatter plot then a fit, graphed with [**matplotlib**]



(a) Graphs of x^2 and 2^x plotted along the original plot



(b) Zoomed in version for the disks vs moves graphs as well as the line for 2^x

Figure 4: Different graphs to establish the relationship between disks and its required moves, as well as other functions to better understand the exponential relationship, graphed with [**matplotlib**]

As seen from Table 1, there is a constant difference between the current term and its preceding term, hinting towards an exponential relationship. Graphically, we can also see it tracing out an exponential pattern, as seen from figure 3 plots a and b.
Finding the exponential relaitionship isn't enough though, as x^2 and 2^x are still both valid assumptions as the difference is always an even number. In order to better investigate this, I plotted the graphs of x^2 and 2^x alsongside the original scatter plot. As demonstrated from figure 4a, it is the 2^x that best approximate the result, while the x^2 graph diverge

significantly as x gets larger. There are, however, discrepency between the expression of x^2 and the actual moves, so figure 4b was created to decrease the range of y values and better understand what the discrepency was. As seen, the difference is a constant of 1, thus deriving the expression for the minimal moves with n disk being:

$$2^{a_{n-1}} - 1 \tag{1}$$

2.2.2 Another Method

There is another method to find the expression1, which is through observing the difference. As seen, the difference is not a constant, thus defeating the idea of x^2. This can seen through its deriative, more specifically the power rule.
For equation x^2, the rate of change is seen through the following:

$$f(x) = x^2 \tag{2}$$

$$\frac{d}{dx} f(x) = x \tag{3}$$

$$\frac{d^2}{d^2x} = 1 \tag{4}$$

This tells us that the second order deriative (the change of the change) will always be one, which is valid.
This is not true, however, for the equation 2^x, as shown in the following:

$$f(x) = 2^x \tag{5}$$

$$\frac{d}{dx} f(x) = 2^{x-1} \tag{6}$$

$$\frac{d}{dx} f(x) = 2^{x-2} \tag{7}$$

As seen, the change will converge to a constant, in fact, the pattern is still the same. When verifying with table 3, it is trivial to see that the change of change does not converge to a number, and since the change is always a power of 2, expression 1 must hold true.
This can also be understood from the perspective of computer programming. As seen from the appendix, the problem is recursive in nature, requiring you to first solve n-1 disks first. Since the process is to first move n-1 disk to the auxiliary peg, then move the n-1 disks to the final peg, the problem will require 2^n steps. However, there the final peg only has to be move once, thus eliminating one additional move, arrive to expression 1.

3 Reflection and Conclusion

As previously mentioned, it was said that 64 disks were used to test the patience of a young priest, below is assessing how much time it would actually take the young priest assuming he can move a disk at a rate of one second per disk, and that he is very familiar with the puzzle, and can thus perform it in the minimal move. As seen from Appendix A, it would take about 580 billion years to complete the task, which is about 42 times the age of the universe (assuming the universe is 13.8 years according to the Lambda-CDM model) [2]. This is clearly not feasible.

Overall, this was a really insightful puzzle to solve, as it involves one of the most fundamental concepts in mathematics and computer science: recursion. During the process of solving this puzzle, I became a better thinker and became better at finding the pattern, thus being able to code this into the computer to help me find the minimal move. This, however, is a great exercise for younger kids, with multiple videos online. It would be insightful to understand how this puzzle help younger kids develop patience and pattern finding and its impact on their lives and future.

References

[1] Science Buddies and Sabine Brabandere. *The Tower of Hanoi*. 2017.

[2] Frank H Shu. *Univser*. 2020.

[3] The Editors of Encyclopaedia and Gupta Kanchan. *Recursive functions*. 2008.