Investigating the Tower of Hanoi

Emma Thomas

December 11, 2013

1 Introduction

The Tower of Hanoi is a puzzle in which a pile of disks, arranged from largest at the bottom to smallest at the top, must be moved from the first peg to the last peg, whilst the following conditions are met[5]:

- Disks must be moved one at a time
- Only the disk at the top of the pile can be moved
- At no point must a larger disk be placed onto a smaller disk

I will be investigating the classic Tower of Hanoi in which a varying number of disks are used with 3 pegs. Therefore the puzzle is complete when the disks are arranged on the third peg starting from largest to smallest, and the aim is to do so in as few moves as possible.

2 Coding for Hanoi Tower in sage

The following code can be used in sage to calculate the minimum number of moves required to complete the Hanoi Tower:

```
def towers(n, source, destination, spare):
    """
    Function to calculate the minimum number of moves using recursive approach.
    Arguments: n = number of disks, source = the first peg,
    destination = the last peg, spare = the middle peg
    Outputs: The minimum number of moves required and a description of the moves.
    """
    count = 0
    if n == 1:  # Checks the base case
        print "Move from peg %s" % source, "to peg %s" % destination
        return 1
    else:
        count += towers(n-1, source, spare, destination)
        count += towers(1, source, destination, spare)
        count += towers(n-1, spare, destination, source)
        return count
print "The minimum number of moves is %s." % towers(3, 1, 3, 2)
```

This code is a modified version of a piece code taken from `http://stackoverflow.com/questions/15052704/how-to-create-a-counter-inside-a-recursive-function`[1].

This code gives the minimum number of moves required and a model solution to the problem as its output:

```
Move from peg 1 to peg 3
Move from peg 1 to peg 2
Move from peg 3 to peg 2
Move from peg 1 to peg 3
Move from peg 2 to peg 1
Move from peg 2 to peg 3
Move from peg 1 to peg 3
The minimum number of moves is 7.
```

2.1 Explaining the code

Figure 1 demonstrates the Hanoi tower in which n disks are red and the blue disk represents n+1. It takes T_n moves to move the pile of n disks from peg 1 to peg 2 (Figure 2), since T_n is the minimum number of moves required to move n disks to another peg. One move is taken to move the largest disk, n+1, from peg 1 to peg 3 (Figure 3). Then T_n moves are required again to move the remaining n disks to peg 3 hence completeing the puzzle as shown in Figure 4.

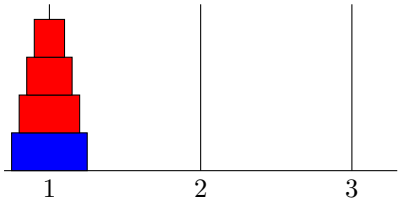
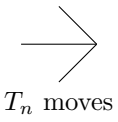


Figure 1: Step 1



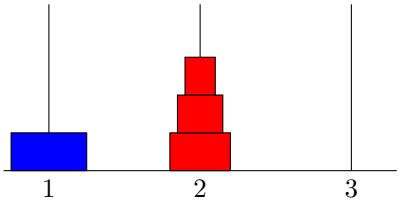


Figure 2: Step 2

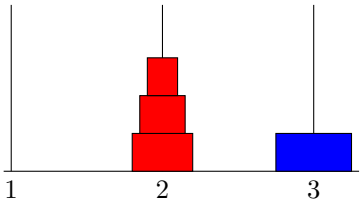
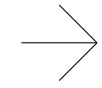


Figure 3: Step 3



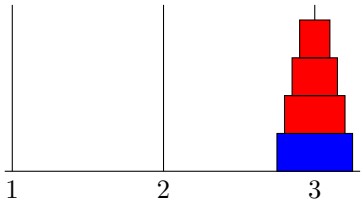


Figure 4: Step 4

Therefore, by the recursive approach, the following equation is given:

$$T_{n+1} = 2T_n + 1 \tag{1}$$

Equation (1) can be used to derive the following fuction:

```
def hanoi(n):
    """
    A function to calculate the minimum number of moves required using recursive approach.
    Arguments: n = number of disks
    Outputs: minimum number of moves
    """
    if n == 1:  # Checks the base case
        return 1
    return 2 * hanoi(n - 1) + 1
hanoi(3)
```

2.2 General Formula

Formula is given by:

$$T_n = 2^n - 1 \tag{2}$$

The following code demonstrates how a function uses equation (2) to calculate the minimum number of moves required to complete the hanoi tower:

```
def moves(n):
    """
    A function to calculate the minimum number of moves required to solve Hanoi Tower using formula.
    Arguments: n = number of disks
    Outputs: number of moves
    """
    return 2 ** n - 1
moves(3)
```

Using this formula, the results for each number of disks used can be easily calculated, as presented in the table:

Number of disks	Minimum number of moves
3	7
4	15
5	31
6	63

3 Graph of Hanoi Tower

Figure 5 shows every arrangement that 3 disks can form, with (1,1,1) meaning all disk are on peg 1 as shown in Figure 6. For example, (3,1,1) represents 2 disks on peg 1, but 1 disk has been moved to peg 3. Therefore, the right-hand side of the triangle in Figure 5 demonstrates the 7 moves required to complete the puzzle in the minimum number of moves possible.

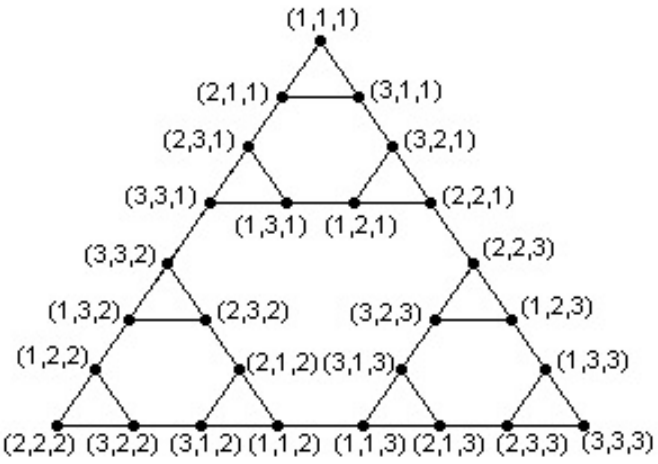


Figure 5: Graph representing Hanoi Tower with 3 disks[3]

Figure 6: Position of disks at (1,1,1) in Figure 5[2]

3.1 Sierpinski triangle

There is a connection with the graph of Hanoi Tower to the famous fractal, Sierpinski triangle. Sierpinski triangle is formed when a triangle has an area removed from its centre by connecting the midpoints of each of its sides. This creates the space of a smaller trangle inside, and 3 remaining shaded triangles. This process can continue infinitely many times. The fractal shown in Figure 7 corresponds to the Hanoi Tower in which 7 disks are used with 3 pegs. An interesting piece of python code generating a Sierpinski triangle graphic may be found at section the 'ActiveCode 4' on `http://interactivepython.org/runestone/static/pythonds/Recursion/graphical.html`.

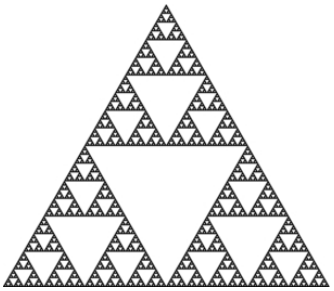


Figure 7: Sierpinski Triangle[4]

References

[1] How to create a counter inside a recursive function, 2013.

[2] Iq puzzles, 2013.

[3] Sierpinski gasket and tower of hanoi, 2013.

[4] Gregory Derfel, Peter J Grabner, and Fritz Vogl. Laplace operators on fractals and related functional equations. *Journal of Physics A: Mathematical and Theoretical*, 45(46).

[5] Wikipedia. Tower of hanoi — wikipedia, the free encyclopedia, 2013. [Online; accessed 11-December-2013].