Speedrunning Tower of Hanoi with Maths

8-20

Bryan Lee Joon Yee 2i2 (8) Leader

Lu Wenkang 2i2 (12)

Wang Guanyu 2i2 (20)

Hwa Chong Institution

(High School)

1 Introduction, Rationale and Background Information

Our group is doing research into the mechanics behind the Tower of Hanoi. The reason why we wanted to do this project is that we found that this game looked very simple, yet was very complicated and challenging when we tried to play it. We felt that there was a potential to research the Tower of Hanoi mathematically.

Tower of Hanoi consists of three rods or towers with *n* disks placed one over the other. The objective of the puzzle is to move the stack to another rod following these simple rules.

1. Only one disk can be moved at a time.
2. No disk can be placed on top of the smaller disk.

As seen from figure 1, this is the desired outcome of the Tower of Hanoi. All of the disks from the left rod are transferred to the right most rod.

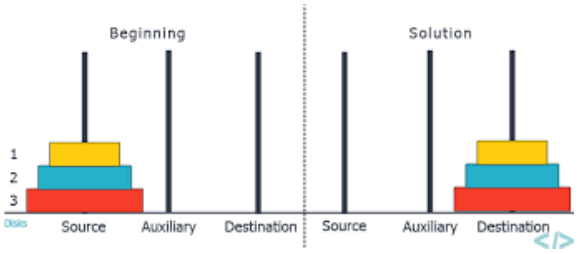


Figure 1: Tower of Hanoi

The Tower of Hanoi has a unique history. There is a story about an ancient temple in India, which has a large room with three towers and 64 golden disks. These disks are continuously moved by priests in the temple. According to a prophecy, when the last move of the puzzle is completed, the world will end. By the way, we do not have to worry about the end of the world yet, because this puzzle requires $2^{64} - 1 = 18, 446, 744, 073, 709, 551, 615$ moves, and even if the priests make 1 move per second, it would take 584, 942, 417, 355 years to complete the puzzle! Then, in 1883, the Tower of Hanoi was introduced by French mathematician Édouard Lucas as a puzzle game. However, what our team was really curious about is why the formula for solving the Tower of Hanoi with the fewest steps is $2^n - 1$? Is there a faster way to solve the Tower of

Hanoi? Are there other formulas which prove this formula? As we ask many questions like these, our project is born - ‘Speedrunning the Tower of Hanoi with Maths’.

2 Objectives

Our first objective is to find out the fastest method to solve the Tower of Hanoi. Next, we would like to find if the same formula can still be applied if some factors are changed. (E.g. Number of disks, Number of rods etc.) Finally, we aim to use programming and simulation to solve the Tower of Hanoi.

3 Research Questions

We have come up with three questions to research on.

1. What is the fastest method to solve the Tower of Hanoi?
2. How will independent variables such as the number of disks and number of rod affect the method to solve the Tower of Hanoi affect the method to solve the Tower of Hanoi?
3. How to solve the Tower of Hanoi with programming?

4 Fields of Math

- Algorithm
- Recursion
- Strategy Research

5 Terminology

Term	**Explanation**
n	Number of disks
r	Number of rods
A, B, C...	Name of the rods corresponding to their positions from left to right (A = 1st from the left, B = 2nd from the left, etc.)
1, 2, 3...	Name of the disks corresponding to the sizes from small to big (1 = smallest disk, 2 = second smallest disk, etc.)
from_rod	The rod where all the disks are at, before any moves are made
aux_rod	Auxiliary rods. The rods where disks are put and are later removed, to be shifted to the to_rod.
to_rod	The destination rod. The rod where all disks have to be stacked for the puzzle to be solved.

Algorithm	An algorithm is a specific and logical procedure to be followed in order to achieve specific results, or to solve a math problem.
Recursion	Recursion is the repeating of the same set of actions. The set of actions is continued or stopped when a condition is satisfied.
Recursive Algorithm	A recursive algorithm is an algorithm which calls itself with "smaller or simpler" input values, and which obtains the result for the current input by applying simple operations to the returned value for the smaller or simpler input.

6 Literature Reviews

Connections in Mathematics: The Tower of Hanoi and the Sierpinski Gasket by Keith Jones, State University of New York, College at Oneonta, n.d.

Keith Jones mentions how the Tower of Hanoi is used by many students to learn about different topics, ranging from mathematics to psychology. It's recursive nature is famous among students studying computer science for learning about recursive functions. This is relevant in our research as it allows us to confirm that coding is a possible method to solving the Tower of Hanoi and thus encouraging us to include it as one of our research questions.

In How Many Steps the kPeg Version of the Towers of Hanoi Game Can Be Solved? By Mario Szegedy, ATT Shannon Labs, Florham Park NJ 07932, USA, 1999

Mario Szegedy talks about a different variation of the popular game, Tower of Hanoi, in which the number of rods is increased, with the rationale being that the original game is much researched and there is not much left to uncover about simply changing the disks. This is similar to our project as we wanted to do a slight extension into different variables and variants of the game due to the original version being heavily researched and not much new to learn about it.

Discussion on Writing of Recursive Algorithm by Song Jinping, Computer Department, Jining Teachers College, Wulanchabu, China, 2013

Song Jinping talks about recursive algorithms in programming, saying that recursion refers to the repetitive phenomenon that the function, process or subroutine calls itself in the running course. He also mentions that recursion is often used in program design, mainly due to the fact that it is simple, concise and can help to reduce the amount of code required. This is relevant to our project because the Tower of Hanoi is tied in very closely with recursive algorithms, and we are also using recursive programming to solve the Tower of Hanoi. Hence we used this to give a simple background to recursive programming and to showcase our rationale behind using recursive algorithms in our coding for solving the Tower of Hanoi.

Generalized Frame-Stewart Numbers by Jonathan Chappelon, Laboratoire de Math´ematiques Pures et Appliqu´ees, Universit´e du Littoral C ̂ote d'Opale and Akihiro Matsuura†, School of Science and Engineering, 2010*

Jonathan Chappelon compares the original version of Tower of Hanoi to the version where the number of rods is varied. He mentions that compared to the original version, the version of Tower of Hanoi in which the number of rods is varied does not have a solution that everyone agrees on, and there appears to be more than one solution possible when finding the minimal number of moves possible. This contributes to our research as it shows that there may be multiple different solutions as compared to our solution when we use coding to test the version of Tower of Hanoi with a different number of rods.

7 Methodology

To achieve our objectives and answer our research questions, we have come up with the following methods.

1. To read up on relevant materials regarding the Tower of Hanoi, to find out the history, usage etc. of the Tower of Hanoi.
2. To experiment with different ways of solving the Tower of Hanoi, thus deriving with different formulas for solving the Tower of Hanoi.
3. To experiment with different variations of the game and investigate if the same formula and method still applies to it.
4. To solve the Tower of Hanoi with coding.

8 Results

Research Question 1: What is the fastest method to solve the Tower of Hanoi?

As the original Tower of Hanoi only includes 3 rods and 3 disks, there really are very few ways to solve the Tower of Hanoi. These methods are linked to each other and eventually it will reach the stage whereby only 1 method is really being used to solve the Tower of Hanoi. The following is a table displaying the steps to solving the classic Tower of Hanoi in 7 steps.

Step 1	Move disk 1 from A to C
Step 2	Move disk 2 from A to B

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

Table 1: Steps for solving classic Tower of Hanoi

So, how can we confirm that 7 is the fewest steps needed to solve the Tower of Hanoi?

Proof that 7 is the minimum number of moves it takes to solve:

To solve the Tower of Hanoi with the fewest steps, we need to move Disk 1 and Disk 2 from Rod A to Rod B, move Disk 3 from Rod A to Rod C, and finally stack Disk 1 and Disk 2 onto Disk 3.

To place Disk 1 and Disk 2 onto Rod B, we need to first move them onto separate rods, Rod C and Rod B respectively, which takes 2 moves as shown from Figure 2 to Figure 4.

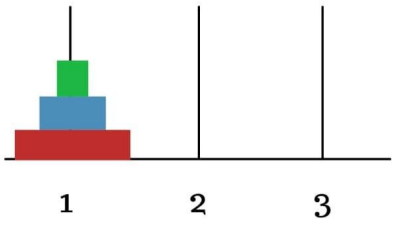


Figure 2: Step 0 for solving classic Tower of Hanoi

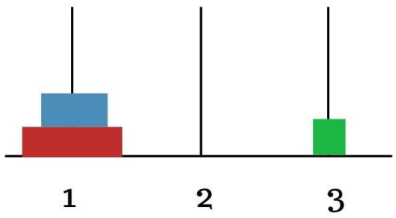


Figure 3: Step 1 for solving classic Tower of Hanoi

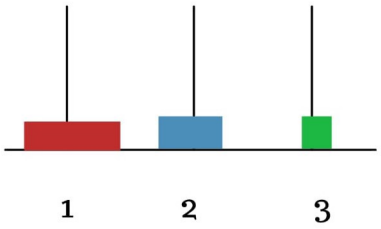


Figure 4: Step 2 for solving classic Tower of Hanoi

Then, move Disk 1 from Rod C to Rod B such that it stacks on Disk 2 so that we can create an empty rod for placing Disk 3 in. This takes 1 move as shown in Figure 5.

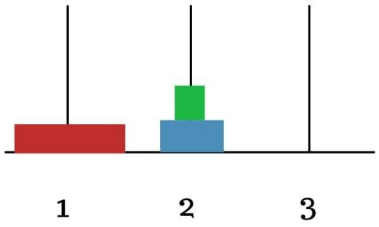


Figure 5: Step 3 for solving classic Tower of Hanoi

Next, Move Disk 3 from Rod A to Rod C as the criteria for solving the Tower of Hanoi is to stack all disks on another rod, with the biggest disk (In this case Disk 3) as the base. This takes 1 move as shown in Figure 6.

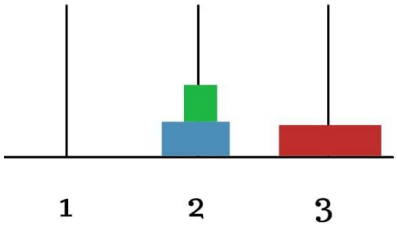


Figure 6: Step 4 for solving classic Tower of Hanoi

Afterwards, move Disk 1 from Rod B to Rod A so that all the disks are on separate disks. This takes 1 move as shown in Figure 7.

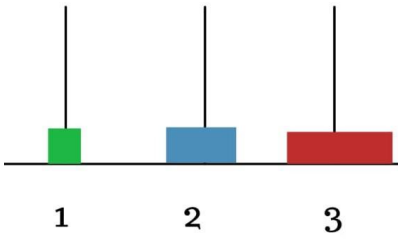


Figure 7: Step 5 for solving classic Tower of Hanoi

Now, we can stack the 2 smaller disks, Disk 1 and Disk 2, onto Disk 3 in Rod C. Move Disk 2 from Rod B to Rod C and finally move Disk 1 from Rod A to Rod C. This takes 2 steps as shown in Figure 8 and Figure 9.

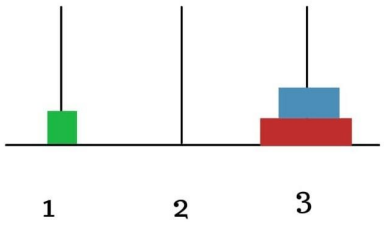


Figure 8: Step 6 for solving classic Tower of Hanoi

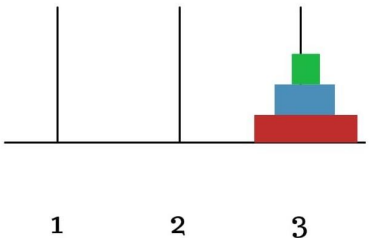


Figure 9: Step 7 for solving classic Tower of Hanoi

Now, let us take a look at how many steps we took in total:

Fewest steps to solve classic Tower of Hanoi =

2 (Move Disk 1 from Rod A to Rod C, Move Disk 2 from Rod A to Rod B)
+ 1 (Move Disk 1 from Rod C to Rod B)
+ 1 (Move Disk 3 from Rod A to Rod C)
+ 1 (Move Disk 1 from Rod B to Rod A)
+ 2 (Move Disk 2 from Rod B to Rod C, Move Disk 1 from Rod A to Rod C)

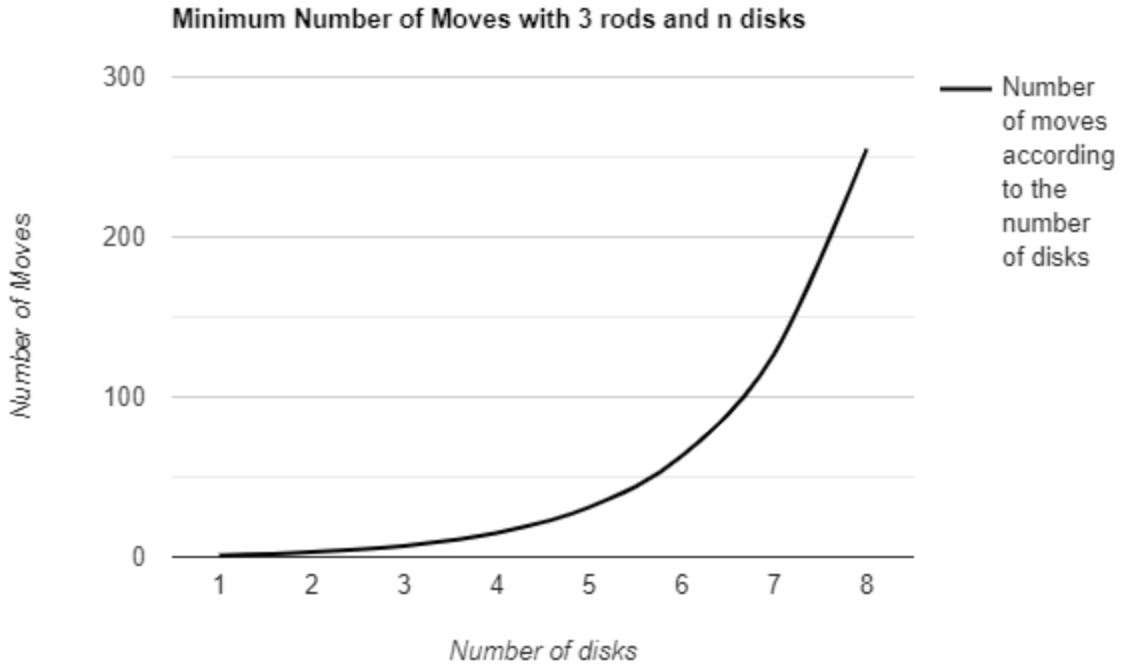
= 7 steps

Thus, from here, we can see that the fewest steps to solving the classic Tower of Hanoi is 7, no matter what.

Research Question 2: How will independent variables such as the number of disks and the number of rods affect the method to solve the Tower of Hanoi affect the method to solve the Tower of Hanoi?

_n_ disks and 3 rods, where _n_ is any natural number

For the puzzle with n disks and 3 rods, where *n* is any natural number, we have the following results.



Graph 1: Different number of disks for 3 rods

Number of disks	**Minimum number of moves required to solve the puzzle**	**Showing that $2^n - 1$ with n as the number of disks is the minimum moves required for the puzzle**
1	1	$2^1 - 1 = 1$

2	3	$2^2 - 1 = 3$
3	7	$2^3 - 1 = 7$
4	15	$2^4 - 1 = 15$
5	31	$2^5 - 1 = 31$
6	63	$2^6 - 1 = 63$
7	127	$2^7 - 1 = 127$
8	255	$2^8 - 1 = 255$

Table 2: Different number of disks

As illustrated in the table and graph above, no matter how many disks there are, the formula $2^n - 1$ still applies, showing that the Tower of Hanoi follows a recursive algorithm, returning to the same formula to derive the minimum number of moves to solve the Tower of Hanoi.

Proof that 2^n - 1 is the minimum number of moves it takes to solve using induction:

Let $M(n)$ be the number of moves it takes to solve n disks given 3 rods.

Claim: $M(n) = 2^n - 1$

Base Case: $n = 1$: $M(1) = 1 = 2^1 - 1$

Assume $M(k) = 2^k - 1$,

$M(k + 1) = 2 \times M(k) + 1$ [$M(k)$ steps for moving all the smaller disks to other rods, 1 step for moving the biggest disk to the last rod, $M(k)$ steps for stacking all the smaller disks onto the last rod]

$= 2\ (2^k - 1) + 1$

$= 2^{k+1} - 1$

Therefore, $M(n) = 2^n - 1$ is the formula for the minimum number of moves required to solve n disks given 3 rods.

Number of rods, $r > 3$

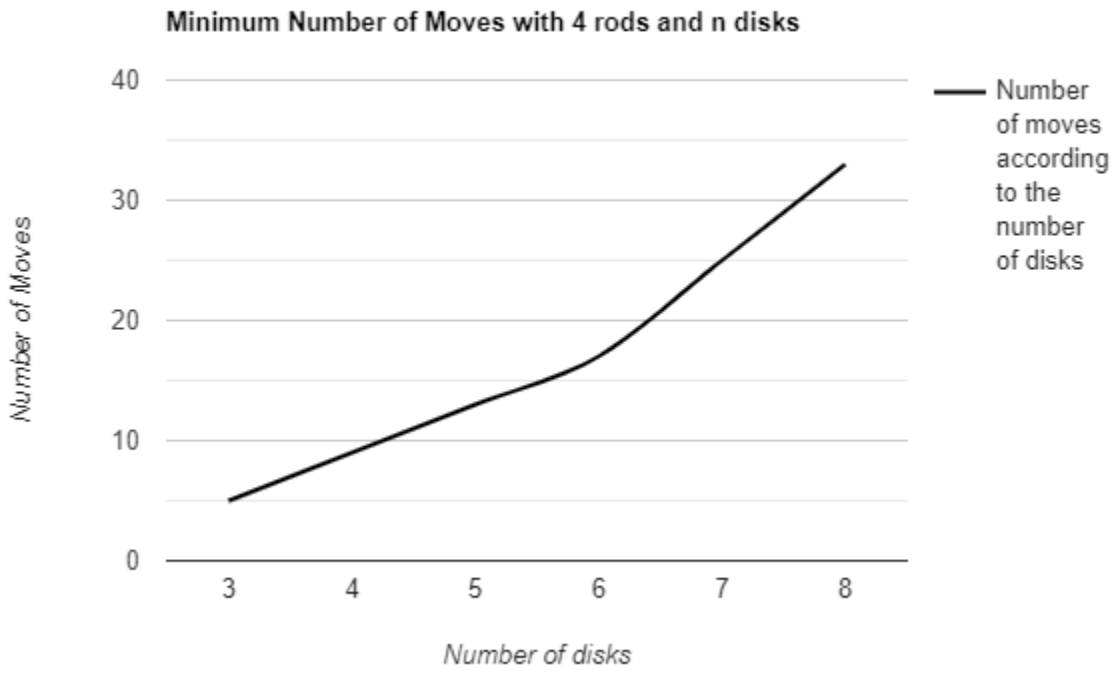
The game has become one of the most common examples of recursive algorithms. It has already been proven that $2^n - 1$ is the formula for the fastest way to solve the Tower of Hanoi (such that the number of rods is always 3), mathematicians started creating other versions of the game, and one of them being to increase the number of rods. Here, we are going to find out the formula of solving the Tower of Hanoi whereby the number of rods is more than 3, and explain the reasoning behind it. As shown in the table below, keeping the number of disks constant in each example as compared to the table above, and increasing the number of rods by 1 to 4. The result is a decrease in the minimum number of moves required to solve the puzzle. For example, 3 disks with 3 rods requires 7 moves to solve, whereas 3 disks with 4 rods requires 5 moves to solve. Our results are shown in the graph and table below.

When number of rods, $r = 4$,

Number of disks	**Minimum number of moves to solve the puzzle**
3	5

4	9
5	13
6	17
7	25
8	33

Table 3: Different number of disks for 4 rods



Graph 2: Different number of disks for 4 rods

In the table below, the number of rods and disks are changed, compared to originally where the number of rods is fixed. The method we used to find the minimum number of moves it takes to solve the puzzle is the Frame-Stewart algorithm, written in Python. This program allows us to input the number of rods and disks, before displaying the minimum number of moves needed to solve the puzzle. This program is universal, and is able to solve all solvable cases. However, this program is not able to display the actual solution to solve the puzzle, as it would be impossible to have a solution for when $r > 2$.

Number of rods	**Number of disks**	**Minimum moves it takes to solve**
3	3	7
5	6	15
6	8	21
7	10	27
8	12	33

Table 4: Differing number of rods and disks

Research Question 3: How to solve the Tower of Hanoi with programming?

Solving Tower of Hanoi using recursive code in Python (3 rods)

We shall illustrate the code using the pseudocode below. Please refer to Appendix A for code.

Pseudocode:

```
create function TowerofHanoi
    if number of disks = 1, move disk 1 from from_rod to to_rod
        Exit
    Run TowerofHanoi with one less disk (Solve top n-1 to aux_rod)
    Move bottom disk to to_rod
    Run TowerofHanoi with one less disk (Solve top n-1 to to_rod)
Set number n
Run TowerofHanoi
```

4 rods recursive solution in Python

We shall illustrate the code using pseudocode below. Please refer to Appendix B for code.

Pseudocode:

```
create function TowerofHanoi
    if number of disks = 0,
        exit
    if number of disks = 1, move disk 1 from from_rod to to_rod
        exit
```

Run TowerofHanoi with one less disk (Solve top n-1 to aux_rod)

Move bottom 2 disks to to_rod

Run TowerofHanoi with n-2 disks (Solve top n-1 to to_rod)

Set number n

Run TowerofHanoi

Minimum moves needed to solve r > 2, Frame-Stewart Conjecture

Please refer to Appendix C for code.

Summarisation of the Frame-Stewart method used in the program:

Frame-Stewart method, whereby *T(n, r)* is the minimum number of moves it takes to transfer n disks using r rods. Take k number of disks, whereby $1 \leq k < n$. Transfer k number of disks to a single rod, other than the start rod and the destination rod, taking *T(k, r)* moves. Then, transfer the remaining disks to the destination rod, taking *T(n - k, r - 1)* moves. Finally, transfer the original top k disks onto the destination rod, taking *T(k, r)* moves. In total, the entire process takes *2T(k, r) + T(n - k, r - 1)* moves. As such, k should be picked when the entire process takes the minimum number of moves. The program runs through every possible case of k, compares the output with the rest, and chooses the smallest output, which is the minimum number of moves to solve that case with n disks and r rods.

9 Conclusion

For research question 1, we have proven that the minimum number of moves to solve the Tower of Hanoi with 3 disks and 3 rods is 7. This sets the basis for research question 2, where we find out the formula to solve the Tower of Hanoi with the minimum number of moves with varying disks.

For research question 2, we first obtained the results for the Tower of Hanoi variant with a standard number of rods, 3, while increasing the number of disks gradually. Using mathematical induction, we can then confirm that $2^n - 1$ is the formula for determining the fastest method, which is to use the least number of moves, to solve the Tower of Hanoi when the number of rods are fixed at 3 and the number of disks are equal or more than 3. We then obtained the results for the variant of the Tower of Hanoi where we increase the number of rods to 4, and increase the number of disks to see the difference in the results from when the rods were fixed at 3 with increasing number of disks. As there is more space to move the disks around as there are more rods, the number of moves required to solve the Tower of Hanoi is reduced as the rods are increased, and our results obtained from experimenting reflected this as well. We then obtained the results using coding for when the number of rods and disks both increase gradually. The results reflected that when the number of disks is equal to the number of rods, the number of moves it takes to solve the puzzle increases linearly at increments of 2, starting from 7 moves when r = 3. However, when the number of disks is one more than the number of rods, the number of moves it takes to solve the puzzle decreases from when r = 3 to r = 4, and then increases again linearly. As such, when r = 3, the number of moves it takes to solve is 15, when r = 4, it takes 13 moves, r = 5 it takes 15 moves, r = 6 it takes 17 moves, and so on. When the number of disks is 1 less than the number of rods, the number of moves it takes to solve the puzzle increases linearly with increments of 2, starting from 3 moves when r = 3.

For research question 3, we used the Frame-Stewart method coded into Python, to derive the minimum number of moves it takes to solve the Tower of Hanoi, where the variables n disks and r rods can be changed independently. This is a universal solution to all solvable cases of the Tower of Hanoi, when $r \geq 3$ and $n \geq 2$.

10 Limitations

There were multiple limitations that we faced throughout the course of this project. One major limitation that affected us this year particularly was the Covid-19 situation, as it caused difficulties in meeting up and discussing, resulting in overall ineffective communication at times.

Furthermore, the Tower of Hanoi is already heavily researched and there is not much space for our own research or any extension, hence we have to resort to mostly proving existing research. Finally, when we did the coding part of our research, we found that a large portion of codes online were either not suited to our use or they were completely obsolete. In the end, we referred slightly to online code and eventually mixed code from online sources with some of our own to get our final code and results.

11 Extensions

These are some other extensions of the Tower of Hanoi we felt would be worth researching if we had the time and resources to as they were interesting. We found out that there are similar puzzles to the Tower of Hanoi like Tower of London, Spin-out etc. which derive from similar concepts as the Tower of Hanoi. We also found other ways of playing the game, like making it a 2 player game, or restricting certain movements. Although we have coded the game in quite a few different languages, there were many more coding languages that we could have potentially researched if there was more time. Finally, we would also have applied the mathematical concept behind the Tower of Hanoi to real life examples such as backup rotation schemes. However, most of our plans fell apart as time did not permit.

12 References

Play Tower of Hanoi. (2018). Play Tower of Hanoi.

Retrieved August 09, 2021, from

https://www.mathsisfun.com/games/towerofhanoi.html

Britannica, T. Editors of Encyclopaedia (2009, July 21). Tower of Hanoi. Encyclopedia Britannica.

Retrieved August 09, 2021, from
https://www.britannica.com/topic/Tower-of-Hanoi

Jones, K. (n.d.). Connections in Mathematics: The Tower of Hanoi and the Sierpinski Gasket, entwined by Self Similar Groups and Finite State Automata.
Retrieved April 05, 2021, from
https://www.oneonta.edu/academics/research/PDFs/LOTM12-Jones.pdf

Mishra, A. (2016, December 26). Tower of Hanoi recursion game algorithm explained.
Retrieved August 09, 2021, from
https://www.hackerearth.com/blog/developers/tower-hanoi-recursion-game-algorithm-explained/

(2016). Binary, Hanoi and Sierpinski, part 1.
Retrieved August 09, 2021, from
https://www.youtube.com/watch?v=2SUvWfNJSsM.

Szegedy, Mario. (1999). In How Many Steps the k Peg Version of the Towers of Hanoi Game Can Be Solved?. 356-361. 10.1007/3-540-49116-3_33.
Retrieved August 09, 2021, from
https://www.researchgate.net/publication/220994907_In_How_Many_Steps_the_k_Peg_Version_of_the_Towers_of_Hanoi_Game_Can_Be_Solved

Song, J. (2013). Discussion on Writing of Recursive Algorithm. Journal Paper Format.
Retrieved August 09, 2021, from
https://gvpress.com/journals/IJHIT/vol6_no6/11.pdf

Chappelon, J., & Matsuura, A. (2011, September 2). On Generalized Frame-Stewart numbers. 1009.0146.pdf.
Retrieved August 09, 2021, from
https://arxiv.org/pdf/1009.0146.pdf.

user8679021user8679021, & Prem SarswatPrem Sarswat 16. (1966, March 1). Tower of HANOI (Frame Stewart) K rods in c. Stack Overflow.

Retrieved August 09, 2021, from

https://stackoverflow.com/questions/46435617/tower-of-hanoi-frame-stewart-k-rods-in-c.

Appendix A

Tower of Hanoi 3 rods n disks code:

```
def TowerOfHanoi(n , from_rod, to_rod, aux_rod):
        if n == 1:
                print("Move disk 1 from rod",from_rod,"to rod",to_rod)
                return
        TowerOfHanoi(n-1, from_rod, aux_rod, to_rod)
        print("Move disk",n,"from rod",from_rod,"to rod",to_rod)
        TowerOfHanoi(n-1, aux_rod, to_rod, from_rod)
n = 3
TowerOfHanoi(n, 'A', 'C', 'B')
```

Appendix B

Tower of Hanoi 4 rods n disks code:

```
def towerOfHanoi(n, from_rod, to_rod, aux_rod1, aux_rod2):
  if (n == 0):                          #If no disk stack to solve, exit
     Return
  if (n == 1):                          #Solve 1 disk stack
     print("Move disk", n, "from rod", from_rod, "to rod", to_rod)
     return
  towerOfHanoi(n - 2, from_rod, aux_rod1, aux_rod2, to_rod)          #Solve for stack of 1 disk
  print("Move disk", n-1, "from rod", from_rod, "to rod", aux_rod2)    #Move bottom disks to to_rod
  print("Move disk", n, "from rod", from_rod, "to rod", to_rod)
  print("Move disk", n-1, "from rod", aux_rod2, "to rod", to_rod)
  towerOfHanoi(n - 2, aux_rod1, to_rod, from_rod, aux_rod2)          #Solve for disk stack to to_rod
n = input(“Input number of disks”)
towerOfHanoi(n, 'A', 'D', 'C', 'B')
```

Appendix C

Tower of Hanoi r rods n disks code:

```
def FrameStewart(ndisks, nrods):
  if ndisks ==0:                        #zero disks require zero moves
```

```
    return 0
  if  ndisks == 1 and nrods > 1:                #if there is only 1 disk it will only take one move
    return 1
  if nrods == 3:                                #3 rods is well defined optimal solution of 2^n − 1
    return 2**ndisks - 1
  if nrods >= 3 and ndisks > 0:
    potential_solutions = (2*FrameStewart(kdisks, nrods) + FrameStewart(ndisks-kdisks, nrods-1) for kdisks in
range(1,ndisks))
    return min(potential_solutions)             #the best solution
  #all other cases where there is no solution (namely one rod, or 2 rods and more than 1 disk)
  return float("inf")
a = int(input("Disks [>] "))
b = int(input("rods [>] "))
print(FrameStewart(a, b))
```