Tower of Hanoi

Eashan Sehdev

Supervised by Professor Andrew Booker
Level M/7
20 Credit Points

March 25, 2025

Acknowledgement of Sources

For all ideas taken from other sources (books, articles, internet), the source of the ideas is mentioned in the main text and fully referenced at the end of the report.

All material which is quoted essentially word-for-word from other sources is given in quotation marks and referenced.

Pictures and diagrams copied from the internet or other sources are labelled with a reference to the web page or book, article etc.

Signed ____________________________

Date 25/03/2025

Abstract

The *Tower of Hanoi* puzzle is a mathematical problem which consists of moving a stack of discs between three or more pegs subject to certain restrictions. Developed in 1882, it has since been studied in depth by mathematicians, both as 'recreational mathematics' and a combinatorial problem.

In this project, we will outline several approaches to the puzzle, present various algorithms, and provide proofs of optimality. A main feature of our investigation will be an algorithm known as the *Frame-Stewart Algorithm* (see §1.3), and we will prove that this results in an optimal solution for 4 pegs. We conclude with a short discussion of the 5-peg problem, and the open problem of whether the Frame-Stewart algorithm produces an optimal solution in general.

Contents

1 Introduction

1.1 History of the Tower of Hanoi puzzle

The Tower of Hanoi puzzle was published in 1882 by the mathematician Édouard Lucas. At the time of its publication, Lucas was said to have “no reputation among professional mathematicians” in France as he only published books on ‘entertaining mathematics’ [5, Page 28]. It was not until 1998 when the mathematician A. M. Décaillot wrote a thesis that brought Lucas’s numerical work to light.
At its core, the puzzle involves moving discs of varying sizes from one peg to another with two simple rules:

- You can only move **one** disc at a time
- You **can not** place a larger disc on top of a smaller disc

The puzzle serves as a classical example of recursion and algorithmic thinking, and one of the possible inspirations is an older problem with a similar solution, the **Chinese rings puzzle**. The earliest known evidence for this puzzle was in the 16th century, in Luca Pacioli’s *De viribus quantitatis*. The puzzle consists of a long loop with a handle on one end that is interlocked with nine rings. The goal is to disentangle the long loop from all nine rings, and the optimal solution takes 341 moves.

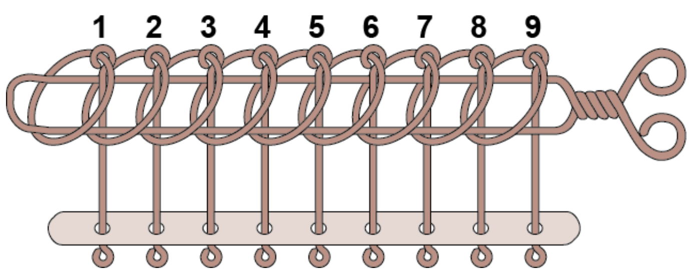


Figure 1: The Chinese Rings Puzzle [1]

Although this puzzle seems complex, it has a fairly simple solution. When there is an odd number of rings, you must simply alternate between these two rules [8]:

Rule A: Ring 1 can always be removed from or placed onto the handle.

Rule B: The only other ring that can be removed from or placed onto the handle is the ring after the lead ring (i.e. the first ring that is still on the handle).

If we use a 3-ring problem, the solution is simply:

1. Apply **Rule A** and take off Ring 1
2. As Ring 2 is now the lead ring, we can apply **Rule B** to take off Ring 3.
3. Apply **Rule A** to put Ring 1 back on the loop.

4. As Ring 1 is now our lead ring, we can apply **Rule B** to take off Ring 2.
5. Apply **Rule A** to remove Ring 1 from the loop.

Now, all the rings have been removed from the loop, completing the solution to the 3-ring problem. As you increase the number of rings, similar to the Tower of Hanoi puzzle, the number of moves required to solve the puzzle grows exponentially.

In order to broaden the appeal of the Tower of Hanoi puzzle, Lucas created a captivating narrative to help market it. He created the Legend of the Tower of Brahma [5, Page 16], which says that under the dome of a temple in Benares, there are three diamond needles on top of a brass plate, each as thick as the body of a bee. God placed 64 gold discs onto one needle, and it was said that if the monks were able to move all the discs from one needle to another whilst following the rules, then the tower, the temple, and the Brahmins will all crumble to dust, and the world will simply vanish. This apocalyptic story was not the only method Lucas used to draw in the attention of the public.

Lucas also claimed that the game was discovered by N. Claus (de Siam)– an anagram of his own name and origin, Lucas d'Amiens. When he published the problem, 'Professor Claus' offered an '*enormous*' amount of money to anyone who could solve the 64 disc puzzle by hand, and that if one move was made every second, it would take over 5 million centuries to complete.

To solve the puzzle for n discs, you must:

- Move $n-1$ discs onto a spare peg
- Move the largest disc onto the target peg
- Move $n-1$ discs from the spare peg to the target peg

With this in mind, Lucas could calculate the number of moves it would take to solve the puzzle for n discs with the recursive solution as such:

1. The total number of moves for n discs, $T(n) = 2(T(n-1)) + 1$
2. Expanding $T(n-1)$, we obtain $T(n) = 2\big[2\,(T(n-2)+1)+1\big]$
3. We can rewrite this as such: $T(n) = 2^2(T(n-2)) + 2 + 1$
4. Repeating this expansion, we get: $T(n) = 2^k(T(n-k)) + 2^{k-1} + \ldots + 2 + 1$ up to $k = n-1$
5. Taking $k = n-1$, we see that the total number of moves for n discs is $2^n - 1$

1.2 Connections with the classical puzzle

The classical 3-peg puzzle has some interesting connections with both **binary** and **geometry**. Starting with binary, we are able to solve the puzzle for any number of discs using just the number of 'trailing zeros' in the binary numbers [9]. A trailing zero is a 0 that follows the first 1 in the binary number.

For example, the 3 disc puzzle would take $2^3 - 1 = 7$ moves to solve optimally. The 7 moves are found as such:

Move Number	Binary	Trailing Zeros Count	Disc Moved
1	001	0	Disc 1
2	010	1	Disc 2
3	011	0	Disc 1
4	100	2	Disc 3
5	101	0	Disc 1
6	110	1	Disc 2
7	111	0	Disc 1

Table 1: Binary numbers 1 to 7 with corresponding disc moves

In order to appreciate the geometric connections, it is crucial to first understand the concept of the Sierpinski triangle. The Sierpinski triangle is a fractal which is formed by taking an equilateral triangle, and recursively removing smaller congruent equilateral triangles from the inside [7]. We are able to use this triangle to map out all possible configurations for the 3-peg puzzle.

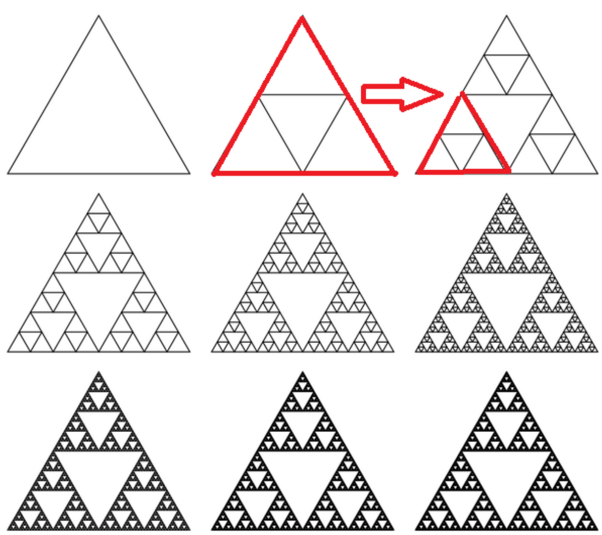


Figure 2: Sierpinski triangle expansion [2]

The top left triangle describes all configurations for the 1 disc puzzle. There are 3 nodes for the 3 configurations, which differ simply by which peg you place the 1 disc. The triangles in

the figure form a sequence from left to right, where each triangle is repeated three times in the shape after it, as highlighted by the drawings in red.
Viewing all configurations for the 3 disc puzzle, we use the top right Sierpinski triangle which can be mapped out as such:

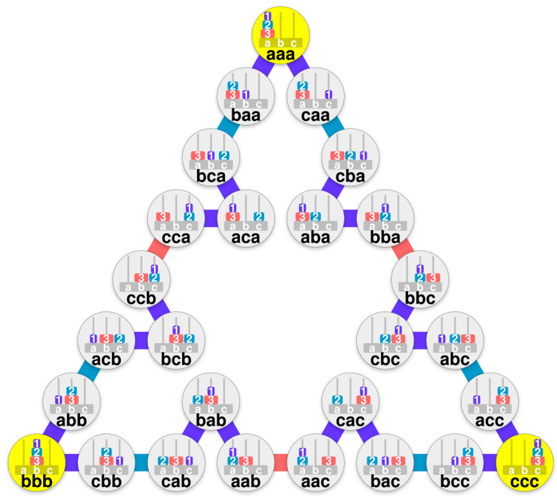


Figure 3: Sierpinski triangle with configurations [10]

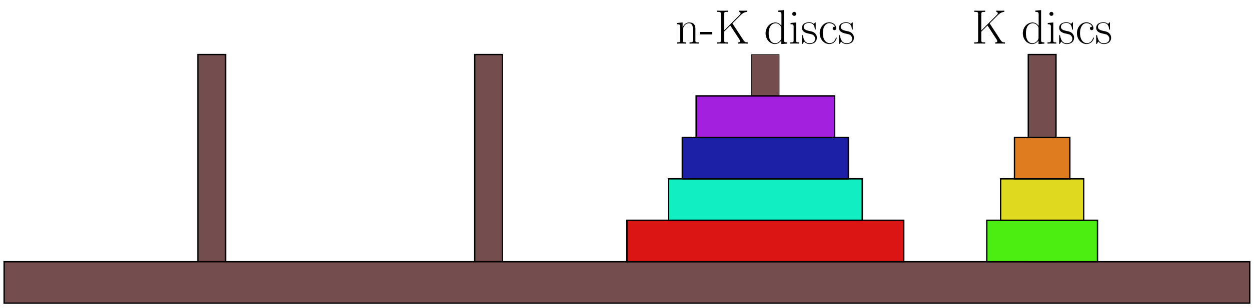
1.3 The Frame-Stewart Algorithm

The puzzle was officially expanded beyond the 3-peg problem in 1907 by Henry Dudeney. He created the Reve's puzzle, which is the Tower of Hanoi puzzle, but with 10 discs across 4 pegs. Increasing the number of pegs complicates the search for the optimal solution. The additional peg means that we now have **two** spare pegs to accommodate the $n-1$ discs, making the problem one of finding the optimal distribution of discs among the spare pegs.

In 1939, the problem was posted in the American Mathematical Monthly journal asking for the solution to the puzzle for n discs across p pegs[5, Page 78]. 2 years later, Frame and Stewart created an algorithm which they claimed was the optimal solution for p pegs. The Frame-Stewart number, denoted $\mathbf{FS}_p^n$, is conjectured to provide the optimal number of moves for the puzzle [5, Page 220]. Focussing on the 4-peg problem, we get the following equation:

$$\mathbf{FS}_4^n = \min\{2\mathbf{FS}_4^K + \mathbf{FS}_3^{n-K}\}$$

Where 'n-K' is in reference to the largest discs, and 'K' is in reference to the smallest discs as pictured below:
$\mathbf{FS}_4^K$ is multiplied by 2 here as these discs are first moved onto a spare peg, using all 4 pegs to do so, and then again moved to the target peg with all 4 pegs at their disposal. We add $\mathbf{FS}_3^{n-K}$, as we can only use 3 pegs to move the largest discs from the source to the target peg. This is because we can't use the fourth peg across, as that would require placing a larger disc on top of a smaller disc which goes against the rules of the puzzle.
In [5, Page 224] we equate n to $\Delta\nu + x$, for $0 \leq x \leq \nu$, and "$\Delta\nu = \frac{\nu(\nu+1)}{2}$". x here serves as the excess between n and the largest triangle number that is less then or equal to n. The



paper suggests that the Frame-Stewart number uses the split for K, where $K = \Delta(\nu - 1) + x$.

As an example of this, if we take $n = 11$, then the largest ν such that $\Delta\nu \leq n$ is 4 (as $\frac{4(4+1)}{2} = 10 \leq 11$ and $\frac{5(5+1)}{2} = 15 > 11$). We equate n to $\Delta 4 + 1$, resulting in $K = \frac{3(3+1)}{2} + 1 = 7$. Consequently, in the 11 peg puzzle, there are 7 'small' discs and 4 'large' discs, giving us as the optimal solution:

$$\mathbf{FS}_4^{11} = \{2\mathbf{FS}_4^7 + \mathbf{FS}_3^4\}.$$

The Frame-Stewart algorithm is the conjectured optimal solution, however in 2014, Thierry Bousch proved it was optimal for the 4 peg problem. This project will go through the proof for the 4 peg problem by Bousch.

2 Foundational proofs

Our goal is to prove the Frame-Stewart algorithm is the optimal solution for the 4-peg problem. Thierry Bousch's paper proves it using this theorem (from [3, Théorème 2.9]):

Theorem 2.1. Take C to be $\{0, 1, 2, 3\}$, $N \in \mathbb{N}$, and $\mathbf{u}, \mathbf{v} : [N] \to C$ as two configurations of N discs. If we assume that two pegs are empty, say 0 and 1, then $\mathbf{v}([N]) \subseteq \{2, 3\}$.
We have the inequality:

$$d(\mathbf{u}, \mathbf{v}) \geq \Psi\{k \in [N] : \mathbf{u}(k) = 0\}. \tag{2.1}$$

The set $\{0, 1, 2, 3\}$ refers to pegs $0, 1, 2$ and 3, u and v refer to the initial and final configurations respectively, and $[N]$ looks at N discs named $\{0, 1, \ldots, N-1\}$. $d(\mathbf{u}, \mathbf{v})$ tells us the total number of moves to go from u to v and the Ψ function will be defined later in this section.

Before we can go into the in-depth proof of this theorem, we need to cover some equations and lemmas that we will use. For $N \in \mathbb{N}$, we define:

$$\Phi(N) := 2^{\nabla 0} + 2^{\nabla 1} + \ldots + 2^{\nabla (N-1)}. \tag{2.2}$$

This is the first equation in Bousch's Paper, and it provides us with an equation for how many moves is required to move N discs. ∇n refers to the triangular root of n, and is the largest possible integer p, where $\frac{p(p+1)}{2} \leq n$. It is also important to note that the triangular number of n, $\Delta n = 1 + 2 + \ldots + n = \frac{n(n+1)}{2}$

We use this equation in order to prove the subsequent equation (from [3, Equation 2.1]):

For all $a, b \in \mathbb{N}$, we have that

$$\Phi(a + b) \leq 2\Phi(a) + 2^b - 1. \tag{2.3}$$

Proof. From Equation (2.2), we see that:

$$\Phi(a + b) = \sum_{n=0}^{a-1} 2^{\nabla n} + \sum_{n=a}^{a+b-1} 2^{\nabla n}.$$

As $\sum_{n=0}^{a-1} 2^{\nabla n}$ is simply $\Phi(a)$, our equation gets rearranged as:

$$\Phi(a + b) = \Phi(a) + \sum_{n=a}^{a+b-1} 2^{\nabla n}.$$

In $\sum_{n=a}^{a+b-1} 2^{\nabla n}$, the sum of the powers of $2^{\nabla n}$ is strictly bounded above by 2^b, as $a \geq 1$, so we have the bound:

$$\sum_{n=a}^{a+b-1} 2^{\nabla n} \leq 2^b - 1.$$

Equivalently, we can rewrite this bound as:

$$\sum_{n=a}^{a+b-1} 2^{\nabla n} \leq \Phi(a) + 2^b - 1.$$

Bringing this all together, we have

$$\Phi(a+b) \leq 2\Phi(a) + 2^b - 1.$$

□

Now we can create an equation for $\Phi(N)$, which tells us the number of moves needed to transfer N discs from one peg to another. To move N discs, for $N > 0$, we must first define an $M < N$ (where M is referred to as $n - K$ in the introduction) so that we can achieve the minimum in the equation.

In the 4-peg problem, the process is as follows:

1. Move the M smallest discs from peg a to a spare peg, denoted peg c, in $\Phi(M)$ moves. (The names for these pegs will be revisited in §3)
2. Use the remaining 3 pegs to move the $N - M$ discs, which will take $2^{N-M} - 1$ moves.
3. Move the M discs onto the target peg in $\Phi(M)$ moves.

This process leaves us with the following equation:

$$\Phi(N) = \min_{0 \leq M < N} \left(2\Phi(M) + 2^{N-M} - 1\right), \quad (N \geq 1). \tag{2.4}$$

We can prove this by showing that there exists a value, M, which provides us with the minimal value for $\Phi(N)$. In order to prove this equation, we must first define a new lemma (from [3, Lemme 2.1]),

Lemma 2.2

If n and p are two natural numbers such that $p \leq n + 1$, we have:

$$\Phi(\Delta n + p) = 1 + (n + p - 1)2^n.$$

Proof. We first define

$$\Psi_N(M) := 2\Phi(M) + 2^{N-M} - 1.$$

We wish then to minimise Ψ_N. In order to find the optimal value of M, we must calculate $\Psi_N(M+1) - \Psi_N(M)$ for different values of M, looking for a change in sign. Using (2.4), we can define these two as:

$$\Psi_N(M) := 2\Big(1 + (m + r - 1)2^m\Big) + 2^{N-M} - 1,$$
$$\Psi_N(M+1) := 2\Big(1 + (m + r)2^m\Big) + 2^{N-M-1} - 1,$$

with the understanding that $M = \Delta m$, and $r \leq m + 1$.
Combining these, we get

$$\Psi_N(M+1) - \Psi_N(M) = 2^{m+1} + 2^{N-M-1} - 2^{N-M}.$$

To simplify this, we recognise that 2^{N-M} is simply $2\big(2^{N-M-1}\big)$, so we are left with:

$$2^{m+1} - 2^{N-M-1}.$$

We have equality with $2^{m+1} = 2^{N-M-1}$ when $M = N - m - 2$.
Using $n = \nabla(N-1)$, we therefore have

$$N - 1 = \Delta n + p.$$

To test for the optimal choice for M, we look at two cases. These are: $M < N - n - 1$ and $M \geq N - n - 1$, and with

$$N - n - 1 = \Delta n + p - n,$$

which is equivalent to $\Delta(n-1) + p$.
Using the fact that $M = \Delta m + r$,
Case 1: If $M < N - n - 1 \Rightarrow \Delta m + r \leq (N - n - 1) - 1$. By subtracting 1 again, we change the strict inequality to a weak one. Hence, we arrive at

$$\Delta m + r \leq \Delta(n-1) + p - 1 \Rightarrow m \leq n - 1.$$

And substituting this back into our earlier equation of $2^{m+1} - 2^{N-M-1}$, we have that $2^{m+1} \leq 2^n$ and $2^{N-M-1} > 2^n$ (as $M < N - n - 1$), so

$$2^{m+1} - 2^{N-M-1} < 0.$$

Case 2: If $M \geq N - n - 1$, then we have $N - M - 1 \leq n$ and $m \geq n - 1$, which leads to:

$$2^{m+1} \geq 2^n \quad \text{and} \quad 2^{N-M-1} \leq 2^n,$$

and yields the result of:

$$2^{m+1} - 2^{N-M-1} \geq 0.$$

As shown, we have a change of sign between the two cases, and so our optimum value for M is obtained at $N - n - 1$.
Hence, $M = N - n - 1$ uniquely minimises $\Psi_N(M)$, and substituting this into $\Psi_N(M)$ gives us

$$\Phi(N) = 2\Phi(N - n - 1) + 2^{n+1} - 1,$$

completing our proof. □

Now we need to define an important result, for $\Psi_L(E)$ defined by

$$\Psi_L(E) := (1-L)2^L - 1 + \sum_{n\in E} 2^{\min(\nabla n, L)}, \tag{2.5}$$

where E is a finite subset of $\mathbb{N}$, and $L \in \mathbb{N}$. We also have that $\Psi(E) = \sup_{L\in\mathbb{N}} \Psi_L(E)$. These results are needed in order to prove the following lemma (from [3, Lemme 2.2]), which defines $\Psi[n]$.

Lemma 2.3

For every natural number n, we have:

$$\Psi[n] := \frac{\Phi(n+1)-1}{2} = \frac{1}{2}\left(2^{\nabla 1} + 2^{\nabla 2} + \ldots + 2^{\nabla n}\right).$$

Proof. We know that $\Psi[0] = 0$ trivially, so we focus on $n \geq 1$. We take $n = \Delta m + p$, where $m = \nabla n$, for any $m \geq 1$ and $0 \leq p \leq m$.
To find $\Phi(\Delta m)$, we use Lemma 2.2 with $p = 0$, giving

$$\Phi(\Delta m) = 1 + (m-1)2^m.$$

Similarly, applying Lemma 2.2 for $\Phi(n+1)$ with $n+1 = (\Delta m + p) + 1$ yields

$$\Phi(n+1) = 1 + (m+p)2^m.$$

For any natural number L, we can compute the difference between $\Psi_{L+1}[n]$ and $\Psi_L[n]$. Using Equation (2.5), we have

$$\begin{aligned}
\Psi_{L+1}[n] - \Psi_L[n] &= (1-(L+1))2^{L+1} - 1 + \sum_{k\in[n]} 2^{\min(\nabla k, L+1)} \\
&\quad - \left((1-L)2^L - 1 + \sum_{k\in[n]} 2^{\min(\nabla k, L)}\right) \\
&= -2L2^L - 2^L + L2^L + \sum_{k\in[n]} 2^{\min(\nabla k, L+1)} - \sum_{k\in[n]} 2^{\min(\nabla k, L)} \\
&= -(L+1) + \sum_{k\in[n]} 2^{\min(\nabla k, L+1)} - \sum_{k\in[n]} 2^{\min(\nabla k, L)}.
\end{aligned}$$

With ∇k being the triangular root of k, and with k ranging from Δn to $\Delta n + 1$, the sum tells us how many k's contribute to the difference in powers of 2 at consecutive triangular root indices, L and $L+1$.
If $\nabla k \geq L+1$, then $2^{L+1} - 2^L = 2^L$, and $2^{\min(\nabla k, L+1)} = 2^{\min(\nabla k, L)}$ for $k \leq L+1$.
Also, for each k where $\nabla k \geq L+1$, the contribution is

$$2^L\Big[\#\{k \in [n] : k \geq \Delta(L+1)\} - (L+1)\Big].$$

We can rewrite this as:

$$2^L\Big[\big(n - \Delta(L+1)\big)^+ - (L+1)\Big],$$

which is strictly positive if and only if

$$\big(n - \Delta(L+1)\big)^+ > L+1.$$

This rearranges to $n > \Delta(L+1) + L + 1$. Using the relationship

$$\Delta(L+2) = \Delta(L+1) + L + 1,$$

we deduce that $n > \Delta(L+1) + L + 1$ implies $n \geq \Delta(L+1) + L + 2$, so $n \geq (L+2)$. Thus, for the expression to be strictly positive, we must have $n \geq \Delta(L+2)$. Since $n = \Delta m + p$ and $n \geq \Delta(L+2)$, it follows that $\nabla n \geq L+2$, and with $\nabla n = m$, we have $L < m-1$. Taking $L = m-2$, we see that $\Psi_L[n]$ reaches its maximum when $L = m-1$ (i.e., when the gap is 0), which maximizes the recursiveness.
Thus,

$$\Psi[n] = \Psi_{m-1}[n] = (2-m)2^{m-1} - 1 + \sum_{k\in[n]} 2^{\min(\nabla k, m-1)}.$$

For $0 \leq k \leq \Delta m$, we have

$$\sum_{0\leq k\leq \Delta m} 2^{\min(\nabla k, m-1)} = \sum_{0\leq k\leq \Delta m} 2^{\nabla k},$$

and for $\Delta m \leq k \leq n$, the minimum value is 2^{m-1}. Therefore,

$$\Psi_{m-1}[n] = (2-m)2^{m-1} - 1 + \sum_{0\leq k<\Delta m} 2^{\nabla k} + \sum_{\Delta m\leq k\leq n} 2^{m-1}.$$

Using Equation (2.2), we obtain

$$\begin{aligned}
\Psi_{m-1}[n] &= (2-m)2^{m-1} - 1 + \Phi(\Delta m) + (n - \Delta m)2^{m-1} \\
&= (2-m)2^{m-1} - 1 + (m-1)2^m + p2^{m-1} \quad (\text{since } \Phi(\Delta m) - 1 = (m-1)2^m) \\
&= (2-m)2^{m-1} - 1 + 2(m-1)2^{m-1} + p2^{m-1} \\
&= \Big(2 - m + 2m - 2 + p\Big)2^{m-1} \\
&= (m+p)2^{m-1} \\
&= \frac{\Phi(n+1) - 1}{2},
\end{aligned}$$

since $\Phi(n+1) = 1 + (m+p)2^m$. $\square$

We can combine this lemma and Equation (2.3), to get the following result for Ψ (from [3, Equation 2.2]):

$$\forall a, b \in \mathbb{N}, \quad \Psi[a+b] \leq 2\Psi[a] + 2^{b-1}. \tag{2.6}$$

To show this holds, we start with the left hand side first:

$$\begin{aligned}
\Psi[a+b] &= \frac{\Phi(a+b+1)-1}{2} && \text{(from Lemma 2.3)} \\
&\le \frac{(2\Phi(a+1)+2^b-1)-1}{2} && \text{(from Equation (2.3))} \\
&\le \frac{2\Phi(a+1)+2^b-2}{2} \\
&\le \Phi(a+1)+2^{b-1}-1.
\end{aligned}$$

The right hand side is equivalent to:

$$2\Psi[a]+2^{b-1} = (\Phi(a+1)-1)+2^{b-1}. \quad \text{(from Lemma 2.3)},$$

As both sides of the equation are equivalent, the inequality holds.

The next lemma (from [3, Lemme 2.3]) provides us with a lower bound for $\Psi[n+2]$.

Lemma 2.4

For all natural numbers, n, we have $\Psi[n+2] \ge 2^{(\nabla n)+1}$.

Proof. If we take $s = \nabla n$, since $\Psi[\cdot]$ is increasing, it is enough to show that

$$\Psi\left[\Delta s+2\right] \ge 2^{s+1}.$$

For $s = 0$ and $s = 1$, we can show that this inequality is actually an equality. Using Lemma 2.3, we have

$$\Psi\left[2\right] = \frac{\Phi(3)-1}{2}.$$

By Equation (2.2),

$$\Phi(3) = 2^{\nabla 0} + 2^{\nabla 1} + 2^{\nabla 2} = 2^0 + 2^1 + 2^1 = 5.$$

Thus,

$$\frac{\Phi(3)-1}{2} = \frac{5-1}{2} = 2.$$

Next, we find $\Psi\left[3\right]$. Using the fact that

$$\Phi(4) = 5 + 2^{\nabla 3} = 9,$$

we obtain

$$\Psi\left[3\right] = \frac{\Phi(4)-1}{2} = \frac{9-1}{2} = 4.$$

This completes our proof for $s = 0, 1$, since $\Psi\left[2\right] \ge 2$ and $\Psi\left[3\right] \ge 4$.
It remains to show that the lemma holds for $s \ge 2$. Note that

$$\Psi\left[\Delta s+2\right] = \frac{\Phi(\Delta s+3)-1}{2} = \frac{\left(1+(s+2)2^s\right)-1}{2} = (s+2)2^{s-1}.$$

Since

$$(s+2)2^{s-1} \geq 4 \cdot 2^{s-1} = 2^{s+1},$$

the desired inequality follows.
(Recall that, for $N = \Delta n + p$ with $0 \leq p \leq n+1$, we have $\Phi(N) = 1 + (n+p-1)2^n$.) $\square$

The next lemma (from [3, Lemme 2.4]) will provide us with a minimum and maximum value for $\Psi(E)$, based on the number of discs, n. We take the upper bound of $2^n - 1$ as the 4 peg problem can always be solved in less than or equal moves to the same number of discs but with 3 pegs.

Lemma 2.5

For every finite subset E of $\mathbb{N}$, we have:

$$n \leq \Psi[n] \leq \Psi(E) \leq 2^n - 1$$

where n is the number of discs, also known as the cardinality of E.

Proof. **The first inequality** immediately follows from:

$$n = \Psi_0[n] \leq \Psi[n].$$

Second inequality:
If we view the n elements of E in an increasing sequence, where

$$e_0 < e_1 < \ldots < e_{n-1},$$

and set the restriction that $\forall k,\ k \leq e_k$, then by Equation (2.5),

$$\Psi_L(E) = (1-L)2^L - 1 + \sum_{0 \leq k < n} 2^{\min(\nabla e_k, L)}.$$

Combining this with $k \leq e_k$ (since $\nabla k \leq \nabla e_k$), we arrive at

$$\Psi_L(E) \geq (1-L)2^L - 1 + \sum_{0 \leq k < n} 2^{\min(\nabla k, L)} = \Psi_L[n].$$

As this holds for all L, the proof of the second inequality is complete, giving us

$$\Psi[n] \leq \Psi(E).$$

Third inequality:
For any natural number L, using Equation (2.5),

$$\Psi_L(E) = (1-L)2^L - 1 + \sum_{k \in E} 2^{\min(\nabla k, L)}.$$

Breaking up the minimum between ∇k and L, we obtain

$$\Psi_L(E) \leq (1-L)2^L - 1 + \sum_{k \in E} 2^L.$$

There are n elements in E, so $\sum_{k \in E} 2^L = n2^L$, and we conclude with

$$\Psi_L(E) = (1+n-L)2^L - 1.$$

Taking any integer s, note that $1 + s \leq 2^s$, so using this with the idea that $s = n - L$, we have $2^{n-L} \geq 1 + n - L$. Multiplying both sides by 2^L yields

$$2^n \geq (1+n-L)2^L,$$

and thus,

$$(1+n-L)2^L - 1 \leq 2^n - 1.$$

This concludes our proof with

$$\Psi_L(E) \leq 2^n - 1.$$

□

The next lemma (from [3, Lemme 2.5]) provides us with an upper bound for the difference between the values of Ψ over two finite sets.

Lemma 2.6

Let A and B be two finite subsets of $\mathbb{N}$, we have:

$$\Psi(A) - \Psi(B) \leq \sum_{k \in A-B} 2^{\nabla k}.$$

Proof. Choose L to be a natural number such that $\Psi_L(A) = \Psi(A)$, then

$$\Psi(A) - \Psi(B) \leq \Psi_L(A) - \Psi_L(B),$$

since $\Psi_L(B) \leq \Psi(B)$. Moreover, as $\Psi_L(A \cap B) \leq \Psi_L(B)$, we have

$$\Psi_L(A) - \Psi_L(B) \leq \Psi_L(A) - \Psi_L(A \cap B).$$

Using Equation (2.5), we can simplify $\Psi_L(A) - \Psi_L(A \cap B)$ as:

$$\left[(1-L)2^L - 1 + \sum_{k \in A} 2^{\min(\nabla k, L)}\right] - \left[(1-L)2^L - 1 + \sum_{k \in A \cap B} 2^{\min(\nabla k, L)}\right],$$

which equals

$$\begin{aligned} &\sum_{k \in A} 2^{\min(\nabla k, L)} - \sum_{k \in A \cap B} 2^{\min(\nabla k, L)} \\ &= \sum_{k \in A-B} 2^{\min(\nabla k, L)} \leq \sum_{k \in A-B} 2^L, \end{aligned}$$

by breaking up the minimum between ∇k and L. Concluding our proof. □

The next lemma (from [3, Lemme 2.6]) provides an upper bound which quantifies the change in the function, Ψ, when an element is removed from a subset, A.

Lemma 2.7

Take A to be a subset of $\mathbb{N}$, and let s be a natural number, such that the set $A - [\Delta s]$ has at most s elements. Then, for all a in A, we have:

$$\Psi(A) - \Psi(A \setminus \{a\}) \leq 2^{s-1}.$$

Proof. Let's start by assuming that A is non-empty, so $s \geq 1$. Using the proof from Lemma 2.3, we have

$$\forall L \geq s-1: \quad \Psi_{L+1}(A) - \Psi_L(A) = 2^L \Big[\#\{n \in A : n \geq \Delta(L+1)\} - (L+1)\Big].$$

Since $L \geq s-1$, we have $\Delta(L+1) \geq \Delta s$. Therefore, there are more n's larger than Δs than there are larger than $\Delta(L+1)$, so we obtain

$$\Psi_{L+1}(A) - \Psi_L(A) \leq 2^L \Big[\#\{n \in A : n \geq \Delta s\} - s\Big].$$

As the set $A - [\Delta s]$ has at most s elements, the number of elements greater than Δs will not exceed s. Hence,

$$\Psi_{L+1}(A) - \Psi_L(A) \leq 2^L \Big[\#\{n \in A : n \geq \Delta s\} - s\Big] \leq 0.$$

This tells us that when L is greater than $s-1$, $\Psi_L(A)$ is a decreasing sequence. Thus, there exists some $L \leq s-1$ such that the optimal sequence of moves satisfies

$$\Psi(A) = \Psi_L(A),$$

which leads to

$$\Psi(A) - \Psi(A \setminus \{a\}) \leq \Psi_L(A) - \Psi_L(A \setminus \{a\}).$$

Using Equation (2.5) for the single point $\{a\}$, we have:

$$\Psi_L(A) - \Psi_L(A \setminus \{a\}) = 2^{\min(\nabla a, L)} \leq 2^L \leq 2^{s-1}.$$

This completes our proof. $\square$

This next lemma (from [3, Lemme 2.7]) provides us with an upper bound for the increase to the function, Ψ, when taking a subset, A, and adding s additional elements.

Lemma 2.8

Let n and s be two natural numbers such that $s \geq 1$ and $n \geq \Delta(s-1)$, and let A be a subset of $[n]$. For any elements $b_1, \ldots, b_s$(which are not necessarily distinct) in $\mathbb{N}$, we

have:
$$\Psi(A \cup \{b_1, \dots, b_s\}) - \Psi(A) \leq \Psi\left[n+s\right] - \Psi\left[n\right].$$

Proof. We define
$$A_t := A \cup \{b_1, \dots, b_t\},$$
where $0 \leq t \leq s$. We need to show that

$$\Psi(A_t) - \Psi(A_{t-1}) \leq \Psi\big[n+t\big] - \Psi[n+t-1]$$

for all t, where $1 \leq t \leq s$, and A_{t-1} is $A_t \setminus \{b_t\}$. We first assume that the inclusion of A_{t-1} in A_t is strict.
Using Lemma 2.3, if we define
$$\sigma := \nabla(n+t),$$
we can simplify the right-hand side of the equation as

$$\Psi\big[n+t\big] - \Psi[n+t-1] = 2^{\sigma-1}.$$

The inequality follows from Lemma 2.7 if we are able to show that the set, $A_t - [\Delta\sigma]$, has at most σ elements. To do this, we start by noting:

$$\sigma = \nabla[n+t] \quad \Rightarrow \quad \Delta(\sigma+1) > n+t \quad \Leftrightarrow \quad \Delta\sigma + \sigma \geq n+t.$$

Now, we need to check the cardinality of $A_t - [\Delta\sigma]$, which we find as such:

$$\#\Big(A_t - [\Delta\sigma]\Big) \leq t + \#\Big([n] - [\Delta\sigma]\Big) = t + \big(n - \Delta\sigma\big)^+.$$

This can be rewritten as

$$\max\{t,\, t+n-\Delta\sigma\} \leq \max\{t,\, \sigma\},$$

since $\sigma \geq n+t-\Delta\sigma$.
Now, all that is left is to show that $t \leq \sigma$, which comes simply from:

$$\Delta t - t = \Delta(t-1) \leq \Delta(s-1) \leq n \quad \Rightarrow \quad \Delta t \leq n+t,$$

concluding our proof. □

The next lemma (from [3, Lemme 2.8]) provides us with a lower bound for the sum of $\Psi(A)$ and $\Psi(B)$ in alignment with the size of the union, $A \cup B$.

Lemma 2.9

Let A, B be two finite subsets in $\mathbb{N}$. Then we have:

$$\Psi(A) + \Psi(B) \geq \frac{\Phi(n+3) - 5}{4} = \frac{1}{2}\Psi[n+2] - 1$$

$$= \frac{1}{4}(2^{\nabla 3} + 2^{\nabla 4} + \ldots + 2^{\nabla(n+2)}),$$

where n is the cardinality of $A \cup B$.

Proof. We take $E = A \cup B$, and let $L \in \mathbb{N}$. From Lemma 2.5, we see that

$$\Psi_L[n] \leq \Psi_L(E),$$

and so we have:

$$\Psi(A) + \Psi(B) \geq \Psi_L(A) + \Psi_L(B)$$

(since $\Psi(A)$ is the maximum value of $\Psi_L(A)$),

$$= \Psi_L(A \cap B) + \Psi_L(A \cup B).$$

Using the fact that $A \cup B = E$ (so $\Psi_L(A \cup B) = \Psi_L(E)$) and that

$$\Psi_L(A \cap B) \geq \Psi_L(\emptyset),$$

we obtain

$$\Psi(A) + \Psi(B) \geq \Psi_L(\emptyset) + \Psi_L(E) \geq \Psi_L[0] + \Psi_L[n].$$

We now take

$$n + 3 = \Delta m + p,$$

so that

$$m = \nabla(n + 3).$$

If $m \geq 2$ and $0 \leq p \leq m$, then

$$n \geq \Delta(m - 2).$$

Using Lemma 2.2 (recalling that $\Phi(\Delta n + p) = 1 + (n + p - 1)2^n$), we have:

$$\Phi(n + 3) = 1 + (m + p - 1)2^m,$$

and

$$\Phi\big(\Delta(m - 2)\big) = 1 + (m - 3)2^{m-2}.$$

If we take $L = m - 2$, then by Equation (2.5) we have:

$$\Psi_L[0] + \Psi_L[n] = (1 - L)2^L - 1 + \sum_{0 \leq k < 0} 2^{\min(\nabla k, L)} + (1 - L)2^L - 1 + \sum_{0 \leq k \leq n} 2^{\min(\nabla k, L)},$$

which simplifies to

$$(1 - L)2^{L+1} - 2 + \sum_{0 \leq k \leq n} 2^{\min(\nabla k, L)}.$$

Substituting $L = m - 2$ into this equation, we get:

$$(3 - m)2^{m-1} - 2 + \sum_{0 \leq k \leq n} 2^{\min(\nabla k, m-2)}.$$

We can break up the sum to separate the minimum as follows:

$$(3-m)2^{m-1} - 2 + \sum_{0 \le k < \Delta(m-2)} 2^{\nabla k} + \sum_{\Delta(m-2) \le k \le n} 2^{m-2}.$$

Using Equation (2.2) and the proof for Lemma 2.3, we obtain:

$$\begin{aligned}
&= (3-m)2^{m-1} - 2 + \Phi\big(\Delta(m-2)\big) + (n - \Delta(m-2))2^{m-2} \\
&= (3-m)2^{m-1} - 2 + \Big(1 + ((m-2)-1)2^{m-2}\Big) + \big(n - \Delta m + 2m - 4\big)2^{m-2} \\
&= (3-m)2^{m-1} - 1 + (m-3)2^{m-2} + (p + 2m - 4)2^{m-2} \\
&= (m + p - 1)2^{m-2} - 1 \\
&= \frac{1}{4}\Big((m+p-1)2^m - 4\Big) \\
&= \frac{\Phi(n+3) - 5}{4}.
\end{aligned}$$

This gives us the result that

$$\Psi(A) + \Psi(B) \ge \frac{\Phi(n+3) - 5}{4}.$$

Using Lemma 2.3, we have the final parts of the result as:

$$\begin{aligned}
\frac{\Phi(n+3) - 5}{4} &= \frac{1}{2}\Big(\frac{\Phi((n+2)+1) - 1}{2}\Big) - 1, \\
&= \frac{1}{2}\Psi[n+2] - 1 \\
&= \frac{1}{4}\Big(2^{\nabla 1} + 2^{\nabla 2} + \ldots + 2^{\nabla(n+2)}\Big) - 1.
\end{aligned}$$

Since $\nabla 1 = \nabla 2 = 1$, we have

$$\frac{1}{4}\Big(2^{\nabla 1} + 2^{\nabla 2}\Big) - 1 = 0,$$

so we get the final part of the result:

$$\frac{1}{2}\Psi[n+2] - 1 = \frac{1}{4}\Big(2^{\nabla 3} + 2^{\nabla 4} + \ldots + 2^{\nabla(n+2)}\Big).$$

$\square$

Now that all the lemmas and equations have been defined, we are now able to prove Theorem 2.1 by breaking up the problem into different cases.In Bousch's paper [3], the term **stacking** is used to refer to the configuration of discs as a unique element of C^N, so in the 4-peg problem, there are 4^N possible stackings.

3 The Cases

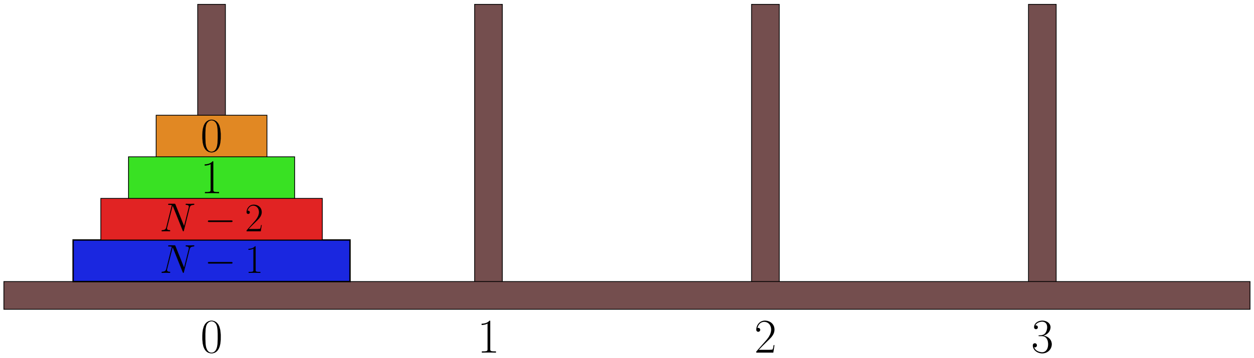
3.1 Notation

Before we can get into the different cases, we must first define some important notation which will help simplify the discussion of configurations in the proofs.

Notation	Definition	Meaning
$[N]$	$[N] = \{0, 1, 2, \ldots, N-1\}$	Labels of the $N \in \mathbb{N}$ discs
C	$C = \{0, 1, 2, 3\}$	Labels of the pegs
$\mathbf{u}$	$\mathbf{u} : [N] \to C$	Configuration at the start (all discs on peg 0)
$\mathbf{v}$	$\mathbf{v} : [N] \to C$	Configuration at the end (all discs across pegs 2 and 3)
D	$d(\mathbf{u}, \mathbf{v})$	Number of moves needed to go from u to v
$\gamma(t)$	γ_t	Configuration at time t
E	$\{k \in [N] : u(k) = 0\}$	Discs initially in peg 0
E'	$\{k \in E : \exists\, t \leq D :\ \gamma_t(k) = 3\}$	Discs that start in peg 0 and pass through peg 3 at least once
T	$\max E'$	Largest disc that starts in peg 0 and passes through peg 3
E''	$\{b_1, \ldots, b_K\} = \{k \in E : k > T\}$	'Large' discs that never go to peg 3 (referred to as $N - M$ in Lemma 2.2)
$\mathbf{x}'$	$\mathbf{x}\|_{[N-1]}$	State of all discs at point x, ignoring disc $N-1$
$\mathbf{x}''$	$\mathbf{x}\|_{[T]}$	State of all discs at point x, ignoring the discs $\geq T$
t_0	$\min\{t : \gamma_t(T) \neq 0\}$	The first time that disc T is not in peg 0
t_1	$\min\{t : \gamma_t(T) = 3\}$	The first time that disc T enters peg 3
t_2	$\min\{t : \gamma_t(N-1) \neq 0\}$	The first time that disc $N-1$ is not in peg 0
t_3	$\max\{t : \gamma_t(N-1) \neq 2\}$	The last time that disc $N-1$ is not in peg 2
$\mathbf{x}_0$	$\gamma(t_0 - 1)$	Configuration of all discs just before T first leaves peg 0
$\mathbf{x}_3$	$\gamma(t_1)$	Configuration of all discs when T first arrives in peg 3
$\mathbf{z}_0$	$\gamma(t_2 - 1)$	Configuration of all discs right before $N-1$ first leaves peg 0
$\mathbf{z}_2$	$\gamma(t_3 + 1)$	Configuration of all discs right after $N-1$ arrives in peg 2 for the last time

Table 2: Notation

The following illustration can be used to help visualise the different pegs.



3.2 Case 1: $E' = \emptyset$

When $E' = \emptyset$, then this tells us that there are no discs which pass through peg 3. This effectively leaves us with the 3-peg problem, and so we know that $D \geq 2^n - 1$, where n is the number of discs in our problem ($n = \#E$). If we use Lemma 2.5, we know that $\Psi(E) \leq 2^n - 1$, so putting this together, we get

$$D \geq \Psi(E),$$

concluding the proof for Theorem 2.1 in this case.

For future cases, we now assume that $E' \neq 0$. If we call the largest element of E', T, then E'' has cardinality K, which can be zero. We refer to the elements of E'' as $b_1, \ldots, b_K$, with $b_1 < b_2 < \ldots < b_K$, and so $E \subseteq [T] \sqcup \{T\} \sqcup \{b_1, \ldots, b_K\}$. (as $[T] = \{0, 1, 2, \ldots, T-1\}$)
As we start labelling the discs from 0, then we have the inequality: $T + K + 1 \leq N$.

Some of the inequalities that we need for the following cases come from Bousch's paper [3], where the induction hypothesis was used. The first is:

$$\begin{aligned} d(\mathbf{u}, \mathbf{z}_0) \geq d(\mathbf{u}', \mathbf{z}'_0) &\geq \Psi\Big(\{k \in [N-1] : \mathbf{u}(k) = 0\}\Big) \\ &= \Psi\Big(E \setminus \{N-1\}\Big). \end{aligned} \tag{3.1}$$

In a similar way, the following inequality is found:

$$d(\mathbf{u}, \mathbf{x}_0) \geq d(\mathbf{u}'', \mathbf{x}''_0) \geq \Psi\Big(E \cap [T]\Big). \tag{3.2}$$

3.3 Case 2: $\Delta K > T$

When taking $E' \neq \emptyset$, the easiest scenario for this is when $\Delta K > T$. This tells us that the Δ of the number of discs considered 'large' exceeds the number for the largest disc which passes through peg 3.

This case implies that $K \geq 1$, and $T \leq b_K = N - 1$. If we look at the set $\{x \in E : x \geq \Delta K\}$, which is a subset of $\{x \in E : x > T\} = E''$. As mentioned above, E'' has cardinality K, so our set of $x \geq \Delta K$ has at most K elements. Combining this with Lemma 2.7, we are able to get the inequality:

$$\Psi E - \Psi(E \setminus \{N-1\}) \leq 2^{K-1}. \tag{3.3}$$

Inequality 3.1 tells us that $d(\mathbf{u}, \mathbf{z}_0) \geq \Psi(E \setminus \{N-1\})$, and we have that $\Psi(E \setminus \{N-1\}) \geq \Psi E - 2^{K-1}$, so combining these results, we get

$$d(\mathbf{u}, \mathbf{z}_0) \geq \Psi E - 2^{K-1}.$$

We can describe the path our stackings take as:

$$\mathbf{u} \ \rightarrow \mathbf{z}_0 \rightarrow \mathbf{z}_2 \rightarrow \mathbf{v}$$

In the configuration of $\mathbf{z}_2$, we have that peg 2 will not contain discs smaller than $N-1$ by definition (since this is the configuration when disc $N-1$ has just arrived here). There will also be no discs smaller than $N-1$ on either peg 0 or peg 1, depending on where the disc $N-1$ moved away from. We refer to this peg as peg c. The discs $b_1, \ldots, b_{K-1}$ will be in $1-c$, i.e. the peg between 0 and 1, which is not c since these discs can't be in peg 3 by definition. If we take c to be peg 0, a possible diagram of this case would look something like:

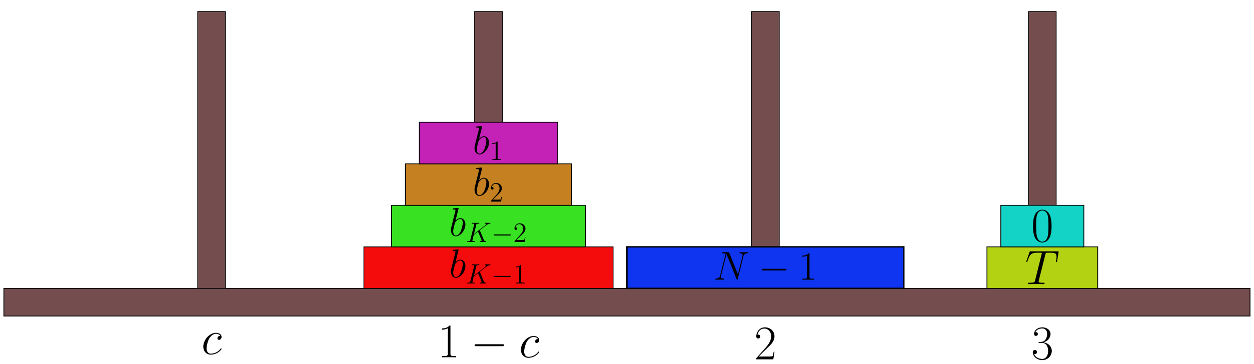


Figure 4: Possible configuration during $\Delta K > T$

The discs $b_1, \ldots, b_{K-1}$ will be on peg 2 in configuration $\mathbf{v}$, and will not have passed through peg 3. Hence, we have that

$$d(\mathbf{z}_2, \mathbf{v}) \geq 2^{K-1} - 1.$$

We know that $d(\mathbf{z}_0, \mathbf{z}_2) \geq 1$, with equality if disc $N-1$ moves directly from peg 0 to peg 2. So, we now know enough to find the total number of moves in this case as

$$\begin{aligned} D = d(\mathbf{u}, \mathbf{v}) &= d(\mathbf{u}, \mathbf{z}_0) + d(\mathbf{z}_0, \mathbf{z}_2) + d(\mathbf{z}_2, \mathbf{v}) \\ &\geq (\Psi E - 2^{K-1}) + (1) + (2^{K-1} - 1) = \Psi E. \end{aligned}$$

This concludes the proof for Theorem 2.1 in this case.

3.4 Case 3: $\Delta K \leq T$

For this case, we have 3 different subcases. The first one is taking $K = 0$, so all the 'large' discs pass through peg 3.

Otherwise, we will have $0 < \Delta K \leq T$, and our 2 subcases differ according to whether disc T enters peg 3 before or after disc $N-1$ enters peg 2 for the last time.
The inequality from the previous case doesn't hold, so we adapt inequality 3.3 using Lemma 2.7 to get

$$\Psi E - \Psi(E \setminus \{N-1\}) \leq 2^{\nabla(T+K+1)-1}.$$

This tells us that $E \setminus [\Delta s]$ has at most s elements, where $s = \nabla(T+K+1)$.

As E is included in $[T+1] \cup E''$, and the difference, $E \setminus [\Delta s]$ contains at most $[T+1-\Delta s]^{+}+K$ elements, it suffices to show that this number is $\leq s$. This is equivalent to showing that $K \leq s$, and $T+1-\Delta s+K \leq s \Leftrightarrow T+K+1 \leq \Delta(s+1)$.

The definition of s tells us that $T+K+1 < \Delta(s+1)$, so our second inequality is automatically met.

To show $K \leq s$, we start by acknowledging that in this case, $K \leq \nabla T$, and we know that $\nabla T \leq s$, so $K \leq \nabla T \leq s \Rightarrow K \leq s$, completing our proof for the above inequality.
Using inequality 3.1, we can get the result:

$$d(\mathbf{u}, \mathbf{z}_0) \geq d(\mathbf{u}', \mathbf{z}_0') \geq \Psi E - 2^{\nabla(T+K+1)-1}.$$

Now taking $K = 0$, which is equivalent to saying that $T = N-1$, then the second largest disc starts in peg 0, ends in peg 2, and passes at least once through peg 3. As $T = N-1$, we have $t_1 \leq t_3$ trivially. The path γ goes through is:

$$\mathbf{u} \to \mathbf{z}_0 = \mathbf{x}_0 \to \mathbf{x}_3 \to \mathbf{z}_2 \to \mathbf{v}$$

Let $c = \gamma t_3(N-1) \neq 2$, so in the $\mathbf{z}$' configuration, we have that pegs 2 and c are empty, and so all the discs are in the other 2 pegs. Viewing the set of pegs $\{0,1,2,3\} - \{2,c\} = \{a,b\}$
To determine what values a, b and c take, we say that $a \in \{0,3\}$ and $b \in \{0,1\}$, with $a, b \neq c$.
The possible combinations are as such:

- If $c = 0$, then a must be 3, and b must be 1
- If $c = 1$, $a = 3$ and $b = 0$
- If $c = 3$, $a = 0$ and $b = 1$

Now we define A and B as such:

$$A := \{k \in [N-1] : \mathbf{z}_2(k) = a\},$$

$$B := \{k \in [N-1] : \mathbf{z}_2(k) = b\}.$$

Then, these sets satisfy $A \sqcup B = [N-1]$, so using Lemma 2.9,

$$\begin{aligned}\Psi(A) + \Psi(B) &\geq \frac{1}{2}\Psi[(N-1)+2] - 1 \\ &= \frac{1}{2}\Psi[N+1] - 1 \\ &= \frac{1}{4}\Big(2^{\nabla 3} + 2^{\nabla 4} + \ldots + 2^{\nabla (N+1)}\Big) \\ &= \frac{1}{2}\Psi[N-1] - 1 + \frac{1}{4}\Big(2^{\nabla N} + 2^{\nabla (N+1)}\Big).\end{aligned}$$

Taking $T = N-1$,

$$\begin{aligned}&= \frac{1}{2}\Psi[N-1] - 1 + 2^{\nabla(N-2)} + 2^{\nabla(N-1)} \\ &= \frac{1}{2}\Psi[N-1] - 1 + 2^{\nabla(N-2)} + 2^{\nabla T} \\ &= \frac{1}{2}\Psi[N-1] - 1 + 2^{\nabla(T-1)} + 2^{\nabla T}.\end{aligned}$$

Taking $K = 0$, it is easy to see:

$$> 2^{\nabla(T+K+1)-1} + \frac{1}{2}\Psi[N-1] - 1.$$

We can use a strict inequality here as increasing n by 1, either increases ∇n by 1, or $\nabla n = \nabla(n+1)$. This means that, for $K = 0$, we have $2^{\nabla(T+K+1)-1} \leq 2^{\nabla T}$, so the two equations differ by <u>at least</u> $2^{\nabla(T-1)}$.

In the configuration $\mathbf{x}_a$, we have that peg a is empty as well as another peg, so we can use the induction hypothesis,

$$d(\mathbf{z}'_a, \mathbf{x}'_a) \geq \Psi A. \quad \text{(by 3.1)}$$

During the configuration $\mathbf{v}'$, we have that pegs 0 and 1 are empty, and as $b \in \{0, 1\}$, we can say, using the induction hypothesis again that

$$d(\mathbf{z}_2, \mathbf{v}) \geq d(\mathbf{z}'_2, \mathbf{v}') \geq \Psi B.$$

Between the configurations of $\mathbf{z}_0$ and $\mathbf{z}_2$, discs that are smaller than the $N-1$ disc, make at least $d(\mathbf{x}'_a, \mathbf{z}'_2)$ moves. Disc $N-1$ makes at least 2 moves in going from peg 0 to peg 3 and ending at peg 2. Hence, we can say that:

$$d(\mathbf{z}_0, \mathbf{z}_2) \geq \Psi A + 2.$$

Putting this all together, we can find the minimum number of moves it takes, D, to complete the puzzle in this case.

$$\begin{aligned}D = d(\mathbf{u}, \mathbf{v}) &= d(\mathbf{u}, \mathbf{z}_0) + d(\mathbf{z}_0, \mathbf{z}_2) + d(\mathbf{z}_2, \mathbf{v}) \\ &\geq \Psi E - 2^{\nabla(T+K-1)-1} + \Psi A + 2 + \Psi B.\end{aligned}$$

Earlier we found that

$$\Psi A + \Psi B \geq 2^{\nabla(T+K+1)-1} + \frac{1}{2}\Psi[N-1] - 1.$$

Putting these two together, we have

$$D \geq \Psi E + 1 + \frac{1}{2}\Psi[N-1] > \Psi E.$$

This completes our proof for Theorem 2.1 in this case with the concluding statement that:

$$D = d(\mathbf{u}, \mathbf{v}) \geq \Psi E.$$

In evaluation of this case, we have that D is strictly greater than ΨE, so taking $K = 0$ is not an optimal way of solving the 4-peg problem.

So now, we assume that $K \geq 1$. This tells us that $T < b_K = N - 1$ and so the largest disc in the puzzle does not pass through peg 3. Now we need to distinguish the order that the puzzle is solved. We compare t_1 with $t_3 + 1$, which is comparing between the first time that disc T enters peg 3 with the (last time that disc $N-1$ is not in peg 2) $+1$, which is equivalent to the time that disc $N-1$ enters peg 2 (assuming that it doesn't leave peg 2, which would never be optimal for the largest disc).

We start by looking at $\underline{t_1 > t_3 + 1}$. This is essentially the case where $\Delta K \leq T$ and disc T enters peg 3 after disc $N-1$ has entered disc 2. The path γ goes through is:

$$\mathbf{u} \to \mathbf{z}_0 \to \mathbf{z}_2 \to \mathbf{x}_3 \to \mathbf{v}$$

A possible configuration for $\mathbf{x}_3$ is as such:

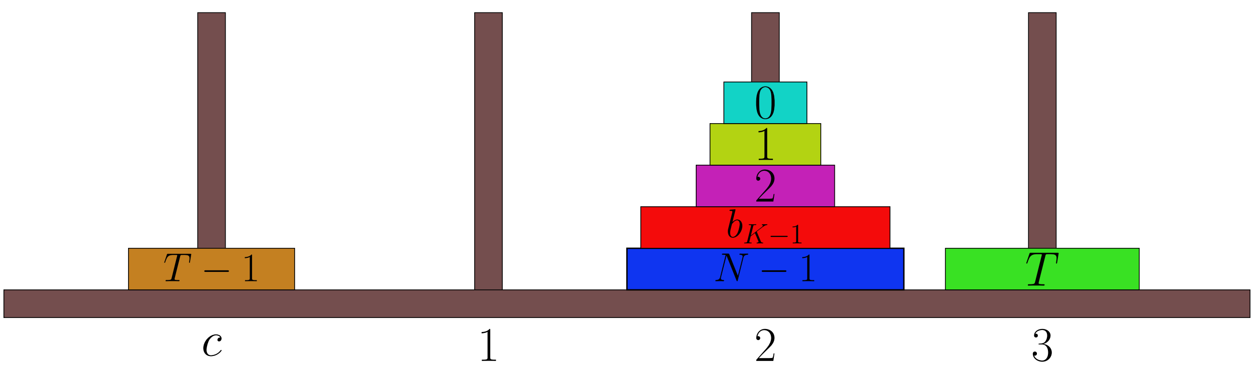


Figure 5: Possible configuration of $t_1 > t_3 + 1$ and $\Delta K \leq T$

Peg 3 has no discs that are smaller than disc T, and the same is the same for the peg which disc T is in under the configuration γ_{t_1-1}, which we will call peg d. As in the other cases, we call peg c, the peg in which disc $N-1$ is during the configuration of γ_{t_3}, and will be $\in \{0, 1\}$
So, we have $\{a, b\} = \{0, 1, 2, 3\} \setminus \{3, d\}$ We take $a \in \{2, c\}$ and $b \in \{0, 1\}$
Looking at possible values for a and b, we have:

- If d is 0, then a must be 2 and b must be 1
- If d is 1, then a must be 2 again, and b must be 0
- if d is 2, then a must be c, and b is simply $1 - c$

Now we define A and B as such:

$$A := \{k \in [T] : \mathbf{x}_3(k) = a\},$$
$$B := \{k \in [T] : \mathbf{x}_3(k) = b\}.$$

Naturally, as pegs a and b are never the same peg, these two sets, A and B are complementary. The configuration $\mathbf{z}_2''$ (The configuration of all discs after $N-1$ arrives at peg 2 ignoring discs larger than or equal to T), will have peg a and another peg as empty. We have the following inequalities:

$$d(\mathbf{x}_3, \mathbf{z}_2) \geq d(\mathbf{x}_3'', \mathbf{z}_2'') \geq \Psi A.$$

(Moving forwards, it is important to understand that $d(\mathbf{x}_3, \mathbf{z}_2) = d(\mathbf{z}_2, \mathbf{x}_3)$) The configuration $\mathbf{v}''$ has pegs b and $1 - b$, i.e pegs 0 and 1 empty, as all discs in $\mathbf{v}$ are split across pegs 2 and 3 by definition. With inequality (3.2), we get the following inequalities:

$$d(\mathbf{x}_3, \mathbf{v}) \geq d(\mathbf{x}_3'', \mathbf{v}'') \geq \Psi B.$$

To go from $\mathbf{z}_0$ to $\mathbf{z}_2$ is at least one move, and in an optimal solution this would most likely be just the one move of disc $N - 1$ going from peg 0 to peg 2. $d(\mathbf{u}, \mathbf{z}_0)$ is the same as in the previous proof $(\Psi E - 2^{\nabla(T+K+1)-1})$.
We now have enough to view the whole journey, D.

$$\begin{aligned} D = d(\mathbf{u}, \mathbf{v}) &= d(\mathbf{u}, \mathbf{z}_0) + d(\mathbf{z}_0, \mathbf{z}_2) + d(\mathbf{z}_2, \mathbf{x}_3) + d(\mathbf{x}_3, \mathbf{v}) \\ &\geq \Psi E - 2^{\nabla(T+K+1)-1} + 1 + \Psi A + \Psi B \qquad \text{(and with Lemma 2.9)} \\ &\geq \Psi E - 2^{\nabla(T+K+1)-1} + \frac{1}{2}\Psi[T+2]. \end{aligned}$$

If we set $s = \nabla(T + K + 1)$. The hypothesis for this case tells us that $\Delta K \leq T$, which is equivalent to $T + K + 1 \geq \Delta(K + 1)$.
Combining this with the definition for s, we get that $s \geq \Delta(K + 1)$. As $T = (T + K + 1) - (K + 1)$, we can now say that $T \geq \Delta s - s = \Delta(s - 1)$. This lets us arrive at $\nabla T \geq s - 1$.
Combining this with Lemma 2.4, we have that $\Psi[T + 2] \geq 2^{(\nabla(\Delta s-1))+1}$ which gives us $\Psi[T + 2] \geq 2^s$.
Combining this with our inequality for D, we have:

$$\begin{aligned} D &\geq \Psi E - 2^{\nabla(T+K+1)-1} + \frac{1}{2}2^s \\ &\geq \Psi E - 2^{\nabla(T+K+1)-1} + 2^{s-1} \\ &\geq \Psi E - 2^{\nabla(T+K+1)-1} + 2^{\nabla(T+K+1)-1} \\ &\geq \Psi E. \end{aligned}$$

Completing our proof of Theorem 2.1 for this case as $D \geq \Psi E$.

Now, the final situation for this case is when $\underline{t_1 < t_3 + 1}$.
In this case, the path γ goes through is:

$$\mathbf{u} \to \mathbf{x}_0 \to \mathbf{x}_3, \mathbf{z}_0 \to \mathbf{z}_2 \to \mathbf{v}$$

We call peg c the peg that contains disc $N-1$ before it arrives at peg 2, and $c \in \{0,1\}$. A possible configuration for $\mathbf{z}_2$ is as such:

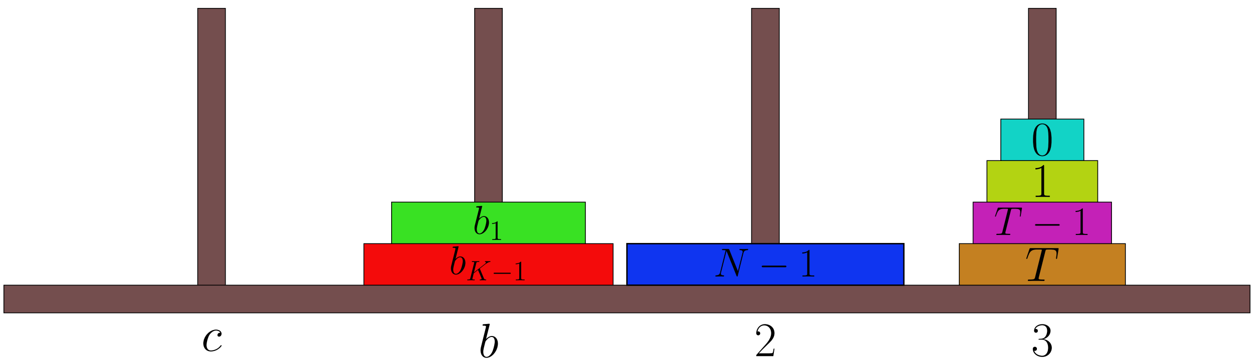


Figure 6: Possible configuration during $t_1 < t + 3 + 1$, and $\Delta K \leq T$

In the configuration for $\mathbf{z}_2'$, we see that both pegs 2 and c are empty. We let b be the peg between pegs 0 and 1 that is not c ($b = 1 - c$). Under the configuration of $\mathbf{z}_2'$, we see that all discs are between pegs b and 3, with discs $b_1, \ldots b_{K-1}$ all in peg b.
In the configuration for $\mathbf{x}_3''$, pegs 3 and another peg are empty. Defining A as:

$$A := \{k \in [T] : \mathbf{z}_2(k) = 3\},$$

we can use inequality 3.2 to get:

$$d(\mathbf{z}_2'', \mathbf{x}_3'') \geq \Psi A.$$

In the configuration for $\mathbf{v}'$, both pegs b and c are empty, i.e pegs 0 and 1. In combination with the induction hypothesis (using inequality 3.2), we get the inequalities:

$$d(\mathbf{z}_2, \mathbf{v}) \geq d(\mathbf{z}_2', \mathbf{v}') \geq \Psi B,$$

where B is:

$$\begin{aligned} B &= \{k \in [N-1] : \mathbf{z}_2(k) = b\} \\ &\supseteq \{k \in [T] : \mathbf{z}_2(k) = b\} \sqcup \{b_1, \ldots, b_{K-1}\}. \end{aligned}$$

We have this inclusion as under the configuration of $\mathbf{z}_2$, we said that discs $b_1, \ldots, b_{K-1}$ were all on peg b.
As all discs are across pegs b and 3 in $\mathbf{z}_2'$, we have that $A \cup B = [T] \sqcup \{b_1, \ldots, b_{K-1}\}$, and so $A \cup B$ has at least $T + K - 1$ elements. We can use Lemma 2.9 to see that:

$$\begin{aligned} \Psi A + \Psi B &\geq \frac{1}{2}\Psi[(T + K - 1) + 2] - 1 \\ &\geq \frac{1}{2}\Psi[T + K + 1] - 1. \end{aligned}$$

Between the configurations $\mathbf{x}_3$ and $\mathbf{z}_2$, the discs that are smaller than T make at least ΨA moves. Between configurations $\mathbf{u}$ and $\mathbf{z}_0$, they make at least $\Psi(E \cap [T])$ moves.
In configuration $\mathbf{u}$, the discs $b_1, \ldots, b_{K-1}$ start in peg 0, and in $\mathbf{z}_0$, they are in another peg (most likely peg 1). As this is done with only 3 pegs, as these discs avoid peg 3, this takes at least $2^{K-1} - 1$ moves from $\mathbf{u}$ to $\mathbf{z}_0$.
We also take at least one move for disc T between $\mathbf{x}_0$ and $\mathbf{x}_3$, and at least one move for disc $N-1$ from $\mathbf{z}_0$ to $\mathbf{z}_2$. We now know enough to set a limit for the number of moves to reach configuration $\mathbf{z}_2$ as:

$$\begin{aligned} d(\mathbf{u}, \mathbf{z}_2) &\geq \Psi A + \Psi(E \cap [T]) + (2^{K-1} - 1) + 1 + 1 \\ &\geq \Psi A + \Psi(E \cap [T]) + 2^{K-1} + 1, \end{aligned}$$

and so,

$$\begin{aligned} D &\geq \Psi A + \Psi(E \cap [T]) + 2^{K-1} + 1 + \Psi B \\ &\geq \left(\frac{1}{2}\Psi[T+K+1] - 1\right) + \Psi(E \cap [T]) + 2^{K-1} + 1 \\ &\geq \frac{1}{2}\Psi[T+K+1] + \Psi(E \cap [T]) + 2^{K-1}. \end{aligned}$$

As $E = (E \cap [T]) \sqcup \{T, b_1, \ldots, b_K\}$. In this case, we have $T \geq \Delta K$, and so, using Lemma 2.8 (with $n = T$ and $s - 1 = K$):

$$\Psi E - \Psi(E \cap [T]) \leq \Psi[T+K+1] - \Psi[T],$$

which gives us

$$\Psi E \leq \Psi(E \cap [T]) + \Psi[T+K+1] - \Psi[T].$$

We can coincide this with our inequality for D to get:

$$\begin{aligned} D - \Psi E &\geq \left(\frac{1}{2}\Psi[T+K+1] + \Psi(E \cap [T]) + 2^{K-1}\right) - \Big(\Psi(E \cap [T]) + \Psi[T+K+1] - \Psi[T]\Big) \\ &\geq \Psi[T] - \frac{1}{2}\Psi[T+K+1] + 2^{K-1}. \end{aligned}$$

We can use Equation (2.6) to see that $\Psi[T+K+1] \leq 2\Psi[T] + 2^K$, so our inequality for $D - \Psi E$ is always positive, as $\Psi[T] + 2^{K-1} - \frac{1}{2}(2\Psi[T] + 2^K) = 0$, leaving us with $D \geq \Psi E$, completing the proof for the final case of Theorem 2.1.

4 Conclusion

By going through all the possible ways of solving the puzzle, we have been able to verify that the Frame-Stewart algorithm is the optimal solution for the 4-peg puzzle, as there is no method of getting to the state $\mathbf{v}$ in fewer moves than ΨE.

When it comes to more than 4 pegs, some work has been put into the 5-peg puzzle. Chen and Shen put some work towards this in [4], and have commented that proving the optimality of $\mathbf{FS}_5^n$ is still incomplete. In [6], Donald Knuth made the comment that "I doubt [that] anyone will ever resolve the conjecture; it is truly difficult."

The reason that the conjecture for the 5-peg puzzle is so difficult to prove is due to the fact that we have 3 spare pegs, and so the number of possible configurations rises significantly. For a small number of discs, the conjecture can be proved by brute force, by going through every possible configuration and checking to see if the problem can be solved in fewer moves than the Frame-Stewart number. However, as we increase the number of discs, there becomes too many configurations to check them all, so we need to prove it with a new hypothesis.

Bousch's induction hypothesis can't be used directly here as the final configuration is defined as all discs being across pegs 2 and 3, with usually just two empty pegs. This hypothesis works for the 4-peg problem as every optimal solution will pass through the state, $\mathbf{v}$. However, with the 5-peg problem, it is not necessarily the case that every optimal solution will pass through $\mathbf{v}$, so a new hypothesis is needed.

With the dramatic increase in computing power over recent years, it is conceivable that the optimality of the Frame-Stewart algorithm for the 5-peg problem can eventually be proven. The immense computing power can be used to check all possible solutions to the puzzle in a reasonable amount of time for a large number of discs, which can help find a configuration that every optimal solution must pass through. Currently there is little evidence to suggest that the Frame-Stewart algorithm is not the optimal solution for 5 or more pegs.

A Python Implementation

Here is a code for the 4-peg problem. It provides the optimal solution for n discs, both as text as well as a graphical display of the step by step solution:

```
import time
import os
import functools

def clear_screen():
    os.system('cls' if os.name == 'nt' else 'clear')          #This command clears the screen to reset the animation between runs.

def render_state(state, disc_max):                         #This helps create a drawing that is easy to read, by ensuring the discs don't overlap between pegs.
    num_pegs = len(state)
    peg_width = 2 * disc_max + 1                    #Width of each peg is larger than the largest disc of the puzzle. Avoids pegs overlapping.
    height = max(len(peg) for peg in state)        #All the pegs remain taller than the total height of the discs in each peg
    lines = []          #Creates a list which gets added to by each line, which we print later on.
    for level in range(height, 0, -1):
        line = ""        #Starts with a blank line which we add to in each step of the puzzle.
        for peg in state:
            if len(peg) >= level:         #Checks if the peg has any discs on it.
                disc = peg[level - 1]       #Takes the disc at the current level.
                disc_width = 2 * disc - 1       #Calculates the width of the disc.
                spaces = (peg_width - disc_width) // 2          #Used to center the disc to the peg.
                disc_str = " " * spaces + "*" * disc_width + " " * spaces     #Used to help visualise the discs with *s.
            else:
                #For empty pegs, this part stills draws a vertical line to represent the empty peg.
                spaces = (peg_width - 1) // 2
                disc_str = " " * spaces + "|" + " " * spaces    #Visualises the empty peg.
            line += disc_str + "   "
```

```
        lines.append(line)    #Adds this line to the others for the
            visualisation.

        #Creates the base line underneath the pegs, with the numbers
            for them.
    base_line = "    ".join(str(i).center(peg_width) for i in range(
        num_pegs))
    lines.append(base_line)    #Adds the horizontal base line to the
         list of lines.
    return "\n".join(lines)    #Gives the visualisation for all the
        lines put together.

def simulate_moves(num_discs, moves, delay=0.5):
    num_pegs = 4    #This code is for the 4-peg problem.
    state = [[] for _ in range(num_pegs)]    #Starts with 4 empty
        lists, one for each peg.
     # Initialise the pegs with all discs in descending order.
    state[0] = list(range(num_discs, 0, -1))
    clear_screen()
    step = 0  #Clears screen, starts at step 0.
    print(f"Initial State (step {step}):")
    print(render_state(state, num_discs))   #Shows the current state
         of the pegs.
    time.sleep(delay)    #Pauses for 0.5 seconds to help with
        visualisation.
    for src, dst in moves:   #Each step works off of the previous
        step.
        disc = state[src].pop()  #Removes the smallest disc from our
             'source' peg.
        state[dst].append(disc)   #Places the moved disc onto the '
            destination' peg.
        step += 1    #increments our step count.
        clear_screen()    #clears the screen for the animation.
        print(f"After move {step}: Move disc from peg {src} to peg {
            dst}")  #Provides text explaining what happens at each
            step.
        print(render_state(state, num_discs))
        time.sleep(delay)   #Pauses 0.5 seconds between moves.
    print("Solved!")

def classical_hanoi(n, source, target, spare):
    if n == 0:
        return []    #If there are no discs, then no moves are
            needed.
    if n == 1:
        return [(source, target)]    #In a one disc puzzle, we move
            the disc to the target peg in one step.
         #This moves all but the largest discs to a spare peg, we
```

```
           move the largest disc over to our target peg, and then
           add all the other pegs to the target.
    return (classical_hanoi(n - 1, source, spare, target) +
            [(source, target)] +
            classical_hanoi(n - 1, spare, target, source))

@functools.lru_cache(maxsize=None)     #Makes the program faster by
   remembering previous moves, and with no limit to its memory of
   moves.
def hanoi4(n, pegs):
      # Provides the optimal solution for n discs with 4-pegs by the
          Frame-Stewart algorithm.

    if n == 0:
        return []   #No discs means no moves again.
    if n == 1:
        return [(pegs[0], pegs[3])]   #One disc, one move from start
            to target.
    best_moves = None   #Stores optimal list of moves.
    best_count = float('inf')  #Starts with allowing for infinite
       moves, and gets updated as we find solutions with fewer moves
       .
   # For the 4-peg problem, we expect pegs = (source, spare, middle,
       target), however we swap the target and middle pegs later in
      the code.
    source, pegB, pegC, target = pegs
    #We Try different splits: move top k discs to peg B, move
       remaining n-k discs (3-peg move) from source to target, then
       move the k discs from peg B to target.
    for k in range(1, n):
        # Moves1: move k discs from source to pegB using all 4 pegs.
        # We choose an ordering that uses target and pegC as spares
           in this step.
        moves1 = hanoi4(k, (source, target, pegC, pegB))
        # Moves2: move remaining n-k discs from source to target
           using classical 3-peg solution (pegC as spare here).
        moves2 = classical_hanoi(n - k, source, target, pegC)
        # Moves3: move the k discs from pegB to target using all 4
           pegs.
        moves3 = hanoi4(k, (pegB, source, pegC, target))
        total = len(moves1) + len(moves2) + len(moves3)   #Total
           number of moves in this split, calculated at each value
           of k.
        if total < best_count:
            best_count = total    #If we find a solution that is
               better than one previously found, we update our
               best_count.
            best_moves = moves1 + moves2 + moves3
```

```
    return best_moves   #Returns the optimal sequence for these n
       discs.

def main():
    try:
        num_discs = int(input("Enter the number of discs: "))    #
           Prompt to allow us to enter the number of discs.
    except ValueError:
        print("Please enter a valid integer.")     #If you input
           something that is not a number, for example a letter,
           then this line provides you with an error message.
        return

    # Peg configuration: (source, pegB, pegC, target).
    # Here, peg 0 is source, peg 1 is used as the intermediate peg (
       B),
    # peg 2 is the spare (C), and peg 3 is the target.
    pegs = (0, 1, 3, 2)    #We swap the order of peg 2 and 3 so the
       graphical solution aligns with the rest of the project.
    moves = hanoi4(num_discs, pegs)
    print(f"Total moves: {len(moves)}")   #Tells us the total number
        of moves needed.
    input("Press Enter to start the animation...")   #Provides you
       an option to create the step by step animation.
    simulate_moves(num_discs, moves, delay=0.5)   #Starts the
       simulation, with our delay of 0.5 seconds between each move.

if __name__ == '__main__':
    main()    #Used to ensure the code only executes if ran directly
       .
```

References

[1] Chinese nine linked rings. https://chinesepuzzles.org/dev/wp-content/uploads/2012/02/Chinese_nine_linked_rings_diagram.png, 2021. Accessed: 2024-11-13.

[2] Aperiodical. Sierpinski triangle (siermathgb3.png). https://aperiodical.com/wp-content/uploads/2015/12/siermathgb3.png, 2015. Accessed: 2025-03-23.

[3] Bousch, T. La quatrième tour de Hanoï. *Bulletin of the Belgian Mathematical Society - Simon Stevin 21*, 5 (12 2014).

[4] Chen, X., and Shen, J. On the frame–stewart conjecture about the towers of hanoi. *SIAM J. Comput. 33*, 3 (2004), 584–589.

[5] Hinz, A. M., Klavˇ zar, S., and Petr, C. *The tower of Hanoi—myths and maths*, second ed. Birkhäuser/Springer, Cham, 2018. With a foreword by Ian Stewart.

[6] Lunnon, W. F. Correspondence. *The Computer Journal 29*, 5 (01 1986), 478–478.

[7] Odom, J., and Wood, T. Exploration of the sierpinski triangle with geogebra. *North American GeoGebra Journal 6*, 1 (2017). Miami University.

[8] Rasmussen, P., Zhang, W., and Liu, N. *Zhongguo chuantong yizhi youxi (Traditional Chinese Puzzles)*. Sanlian Shudian, 2021. See pages 20–53. In Chinese.

[9] Valholl. Solving tower of hanoi with the rhythm of counting of binary number, 2022. Accessed: 2025-03-03.

[10] Wikipedia. Tower of Hanoi graph. https://upload.wikimedia.org/wikipedia/commons/2/2c/Tower_of_hanoi_graph.svg, 2025. Accessed: 2025-03-22.