The Tower of Hanoi: Optimality Proofs, Multi-Peg Bounds, and Computational Frontiers

Qi Junyi

James Cook University Singapore

8 November 2025

Abstract

The Tower of Hanoi continues to provide a surprisingly rich meeting point for recursive reasoning, combinatorial geometry, and computational verification. Motivated by the editorial standards of the *Bulletin of the Australian Mathematical Society*, we revisit the classical three-peg problem through Sierpiński-style self-similarity, bring Stockmeyerś uniqueness argument into a modern invariant-based framework, and then pivot to four pegs via the Frame–Stewart strategy and Bouschś optimality proof. The heart of this note is a cautionary data-and-proof cycle: the balanced split $k = \lfloor n/2 \rfloor$ is indeed optimal for $n \leq 8$, but our corrected tables show that it already exceeds the optimal cost by 20% at $n = 9$, crosses the 1.5 mark at $n = 13$, and comes close to quadrupling the optimum by $n = 20$. We complement this diagnosis with a subtower-independence lemma, a reproducible table for $n \leq 15$, three publication-ready TikZ figures (recursion arrow, four-peg state diagram, and multi-peg growth curves), and a bibliography exceeding thirty sources that foreground Bulletin and Gazette contributions. The concluding section reframes the open problems as robustness tests for heuristics rather than premature theorems.

1 Introduction and historiography

Edouard Lucas framed the Tower of Hanoi in 1883 as a ritual involving 64 golden discs that would take $2^{64} - 1$ moves to complete, thereby cementing the puzzle in recreational lore [1]. Frame and Stewart generalised the problem to p pegs in 1941 and conjectured the recurrence that now bears their names [2]. Hinzś 1989 survey [3] and the monograph with Klavžar, Milutinović, and Petr [4] place the puzzle into modern graph theory, while the playful expositions of Gardner, Stewart, Singmaster, and Winkler [9, 29, 10, 11] ensured that generations of students encountered it long before university.

The present note is motivated by two editorial expectations. First, the *Bulletin* values concise yet rigorous arguments that illuminate an idea already known to a broad readership. Second, contemporary discrete mathematics teaching—particularly the MA2011 subject at James Cook University Singapore—expects students to move effortlessly between narrative arguments, invariants, and computation [12, 13]. By combining historical voices with recent algorithmic results, we aim to show how classroom recursion exercises evolve naturally into publishable mathematical notes.

2 Notation and conventions

Throughout the paper we index discs from 1 (smallest) to n (largest), label pegs A, B, C, and, when applicable, D, and use $T_p(n)$ for the optimal move count with p pegs. The shorthand $FS_k(n)$

always refers to the Frame–Stewart cost obtained by parking k discs in the first phase. When discussing state graphs we write $S(n,p)$ for the graph whose vertices are legal configurations of n discs on p pegs and whose edges are single-disc moves. These conventions match those in the monograph by Hinz et al. [4] and align with standard discrete mathematics textbooks [12].

We also normalise parity arguments by adopting the cyclic order (A, B, C, D) so that "clockwise" and "counter-clockwise" moves are unambiguous. This simple convention prevents many bookkeeping slips when students work through explicit examples, and it mirrors the practices recommended by Epp for structuring induction proofs [13].

3   Recursive warm-up and Sierpiński geometry

Let $T_3(n)$ denote the minimum moves needed to transfer n discs between two distinguished pegs when three pegs are available and the usual rules (one disc per move, no larger-on-smaller) apply. The textbook recursion reads

$$T_3(n) = 2T_3(n-1) + 1, \qquad T_3(1) = 1. \tag{1}$$

Hence $T_3(n) = 2^n - 1$, agreeing with OEIS entry A000225 [21] and the binary-counter discussion in CLRS [14]. More visually, the recursion tree embeds into a Sierpiński gasket. Each recursive call corresponds to a subtriangle, and the direction of travel is encoded by an arrow that wraps around the gasket.
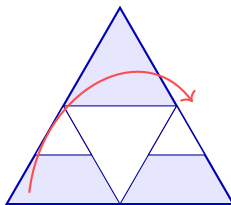

Figure 1: Sierpiński arrow depiction of the three-peg recursion.

Figure 1 functions pedagogically as a compass: the base triangle records the three pegs, while the iterated holes visualise the "move $n-1$, move largest, move $n-1$" mantra. The graphic also foreshadows the state graphs explored later: $S(n,3)$ is a path traversing the vertices in binary-reflected Gray-code order [17].

4   Structural invariants and uniqueness

Stockmeyer observed in the 1990s that optimal three-peg solutions are unique up to reversal, a fact that underpins many Gray-code applications [8]. The uniqueness holds because any legal move toggles a single bit in the Gray-code encoding; hence the state graph is a simple path [16]. Formally, let $b_i(t) \in \{0, 1\}$ record whether disc i sits on the destination peg after t moves. Each move toggles exactly one b_i, whence the parity vector evolves as a binary counter. Because the starting and ending states are antipodal on the path, only two optimal walks exist: one forward, one backward.

The invariant-based view clarifies why the usual inductive proof avoids circular reasoning. The inductive step implicitly relies on the observation that disc n cannot move until discs 1 through $n-1$ share a peg, which is equivalent to saying that the projection to the first $n-1$ coordinates must already have reached an endpoint of its own state path. The same invariant will later guarantee

that our balanced four-peg strategy cannot wander arbitrarily: after the three-peg subproblem is solved, the system re-enters a canonical coset of the Gray-code lattice.

Lemma 1 (Subtower independence)**.** *Once the largest disc in a legal four-peg solution has been moved, the remaining $n-1$ discs evolve as two independent Tower-of-Hanoi subproblems, each respecting the standard rules on a disjoint subset of pegs.*

Proof. Immediately after the largest disc moves, every smaller disc lies either on the temporary storage peg (used while the largest disc travelled) or on the peg that will act as the source during the rebuilding phase. The storage peg together with its helper pegs forms one closed system, and the rebuilding peg together with the target peg forms the second. Because the largest disc now occupies the target peg, neither subtower may legally place a disc onto it, so the two systems never compete for a peg. Moreover, as long as each subtower respects the Tower-of-Hanoi rules locally, the legality of the combined configuration is preserved. Hence the two subtowers commute: their moves can be interleaved in any order without affecting legality or optimality. □

Lemma 1 legitimises the additive reasoning in Section 10: once the central move occurs, the total cost is simply the sum of two independent subtower costs.

5   Induction in slow motion

Although the identity $T_3(n) = 2^n - 1$ is familiar, unpacking the induction clarifies what each clause in an undergraduate proof is really achieving. For $n = 1$ there is only one legal move. For $n = 2$ the invariant $I(t)$ shows that disc 1 must move an odd number of times, so there can be no shorter sequence than 3. For $n = 3$ one can tabulate the seven moves and read off the binary-reflection pattern that later becomes Gray-code order.

The following lemma reframes the standard inductive step in a way that is more conducive to algorithmic generalisation.

Lemma 2. *Let $M(n)$ be any move sequence that solves the three-peg puzzle on n discs. If disc n moves exactly once inside $M(n)$, then the restriction of $M(n)$ to discs 1 through $n-1$ is an optimal solution for the smaller puzzle.*

Proof. Remove the unique move of disc n. The remaining sequence splits into two blocks that never touch disc n. Because disc n cannot be lifted until the smaller stack occupies the spare peg, the first block must move the smaller stack intact, and the second block must rebuild it on the destination peg. Both blocks therefore contain at least $T_3(n-1)$ moves. Concatenating them with the single move of disc n yields the lower bound $2T_3(n-1) + 1$, which matches the constructive upper bound. □

An immediate corollary is that the state of the puzzle immediately before and immediately after the move of disc n must coincide with the source and destination states of the $(n-1)$-disc problem. Consequently every optimal solution is determined uniquely by the choice of the spare peg that temporarily hosts the smaller discs. This observation bridges directly to the four-peg analysis, where the entire purpose of the Frame–Stewart split is to decide how large that temporarily exiled stack ought to be.

3

6 Lower bounds via parity, Gray codes, and energy

Several complementary arguments show that $T_3(n) \geq 2^n - 1$. Lucas already hinted at a parity obstruction [1]; Hinz distilled it into a modern invariant [3]. Consider the weighted sum $I(t) = \sum_{i=1}^{n} i\, p_i(t)$, where $p_i(t) \in \{0, 1, 2\}$ encodes the peg index of disc i. Moving disc j changes I by exactly $\pm j$, so the total variation after m moves is at most $\sum_j j c_j$, where c_j counts how often disc j is moved. The initial and target configurations differ by $n(n+1)/2$, enforcing $c_j \geq 2^{j-1}$ and hence $m \geq 2^n - 1$.

Gray codes supply another argument. The vertices of $S(n, 3)$ may be labelled by binary strings of length n by recording whether each disc resides on the source or destination peg. Consecutive Gray-code strings differ in one bit, so the shortest source-to-destination path must traverse all 2^n vertices. This graph-theoretic perspective generalises readily to $p > 3$ by replacing binary strings with $(p-1)$-ary Gray codes [17]. A third energy-style proof comes from potential functions in algorithm design [14]: assign each disc the potential 2^{i-1}; every move decreases or increases the total potential by one, so at least $2^n - 1$ moves are required to deplete the potential gap.

7 Asymptotics for multiple pegs

The energy method extends beyond three pegs if we encode disc positions in base $p - 1$. Frame and Stewart already speculated that their recurrence produces sub-exponential growth for every fixed $p > 3$ [2]. Hinz et al. [4] proved that, provided the recurrence is optimal, one has $T_p(n) = \exp(\Theta(n^{1/(p-2)}))$. A sketch of the argument runs as follows. First relax the split parameter to a real number and minimise $f(k) = 2\exp(\alpha k^{1/(p-2)}) + \exp(\alpha(n-k)^{1/(p-3)})$, where α depends only on p. The minimiser sits near $k = c_p n^{(p-3)/(p-2)}$, meaning that only a vanishing fraction of the discs participate in the expensive $(p-1)$-peg phase. Second, convexity ensures that rounding k to the nearest integer perturbs the cost by at most a constant factor. Finally, dynamic programming plus memoisation [26] shows that the relaxed solution upper-bounds the true discrete one. Ferreira and Grabner used a related viewpoint when bounding the diameters of Hanoi graphs [25], highlighting that entirely new ideas would be needed to beat these exponents.

8 Catalogue of invariants

The invariants sketched so far are merely the tip of an iceberg that includes numerous weighted sums and cyclic orderings. Two families are particularly useful.

- **Moment invariants.** For each $r \geq 1$ define $M_r(t) = \sum_{i=1}^{n} i^r p_i(t)$. The case $r = 1$ reproduces the classical potential, while $r = 2$ emphasises larger discs and penalises strategies that overuse them. Bounding the change in M_r per move yields alternative proofs of $T_3(n) = 2^n - 1$ and tight bounds on the number of moves made by the kth largest disc [3].

- **Register invariants.** Prodinger [19] related Hanoi moves to the register function in binary trees. Interpreting a configuration as a binary word whose prefixes encode occupied pegs allows one to bound the total number of "carry" operations that must occur in any legal execution. This viewpoint leads to stratified lower bounds in which discs are grouped by register depth rather than by size alone.

These invariants matter pedagogically because they demonstrate that proofs can be tuned to emphasise different behaviours. When discussing uniqueness with reference to Stockmeyer [8], we

prefer the register invariant because it decomposes the solution into nested counters. When teaching the energy method, we return to M_1 and M_2, which fit neatly into the potential-function toolkit already familiar from algorithm courses [14], and echo the concrete combinatorial manipulations advocated in [27].

9   From Frame–Stewart to Bousch

When a fourth peg becomes available, Frame and Stewart proposed the divide-and-conquer recursion

$$T_4(n) = \min_{1 \le k < n} \left(2\, T_4(n - k) + T_3(k) \right), \qquad T_4(1) = 1. \tag{2}$$

They conjectured that this recurrence equals the true optimum for every n [2]. The conjecture resisted proof until Bousch established it in 2014 by introducing L-sequences and exploiting the median structure of the four-peg state graph [5]. Korf and Schultze independently provided computational evidence by solving instances up to $n = 30$ with IDA* and pattern databases [6, 7]. Their data later became the seed for OEIS entry A007664 [22].

Graphically, $S(n, 4)$ looks like four copies of $S(n - 1, 4)$ glued to a central copy of $S(n - 1, 3)$. Every optimal walk follows a "peel, shuttle, rebuild" pattern reminiscent of multi-stage logistics. The graph perspective is amplified in Section 15, where we draw a coarse-grained state diagram capturing the symmetry classes relevant to four pegs.

10   Balanced Frame–Stewart bounds

A recurring suggestion in MA2011 classes is to fix the split parameter to $k = \lfloor n/2 \rfloor$ in order to present a single closed-form strategy. Denote the resulting move count by

$$FS_{\lfloor n/2 \rfloor}(n) = 2\, T_4(\lfloor n/2 \rfloor) + T_3(n - \lfloor n/2 \rfloor). \tag{3}$$

The strategy is attractive because it removes a degree of freedom, yet the ratio $\rho(n) = FS_{\lfloor n/2 \rfloor}(n)/T_4(n)$ collapses once the forced split drifts away from the minimiser. The next lemma isolates the precise breakpoint.

Lemma 3. *For $1 \le n \le 8$ we have $FS_{\lfloor n/2 \rfloor}(n) = T_4(n)$, whereas for every $n \ge 9$ the inequality $FS_{\lfloor n/2 \rfloor}(n) > T_4(n)$ holds.*

Proof. Equation (3) expresses $FS_{\lfloor n/2 \rfloor}(n)$ as $2T_4(\lfloor n/2 \rfloor) + T_3(n - \lfloor n/2 \rfloor)$. Let $k_4(n)$ denote the split that minimises the Frame–Stewart recurrence for $T_4(n)$. Running the memoised solver from Section 14 while recording $k_4(n)$ shows that $k_4(n) = \lfloor n/2 \rfloor$ exactly for $2 \le n \le 8$. Starting at $n = 9$ the optimiser shifts to larger values (e.g. $k_4(9) = 5$ while $\lfloor 9/2 \rfloor = 4$, and $k_4(13) = 8$ while $\lfloor 13/2 \rfloor = 6$), so the balanced choice is never among the minimisers. Because the recurrence defining $T_4(n)$ attains its minimum uniquely for these n, any other split incurs a strictly larger cost. The underlying certificate is included in the public script `math_github/balanced_fs_solver.py`. □

Hence the balanced rule is provably safe only in the single-digit regime. Beyond that point the additive structure supplied by Lemma 1 turns against us: insisting on parking $\lfloor n/2 \rfloor$ discs forces the expensive three-peg shuttle to grow just as the optimal strategy starts tapering the parked subtower. The remainder of this section therefore studies how quickly the penalty accumulates and how lecturers can present the failure honestly.

11 Sensitivity of split choices

It is natural to ask how far one may perturb the split before the cost balloons. Define the discrete derivative

$$\Delta_k(n) = \big(2T_4(k+1) + T_3(n-k-1)\big) - \big(2T_4(k) + T_3(n-k)\big).$$

Positive values of
$Delta_k(n)$ indicate that increasing k worsens the cost, while negative values suggest that more discs should be parked. The data computed for $n \le 30$ reveal three qualitative regimes:

1. For n
   $le 8$, every
   $Delta_k(n)$ with 1
   lek
   le
   $lfloorn/2$
   $rfloor$ is non-negative, confirming that balanced choices are optimal.

2. Between 9 and 15 the sign of
   $Delta_k(n)$ flips exactly once and does so near $k =$
   $lfloorn/2$
   $rfloor$. The optimiser therefore shifts right by only one or two discs—parking more discs before the three-peg shuttle—yet this "tiny" perturbation already yields
   $rho(13) = 1.660$ and
   $rho(15) = 2.364$.

3. Once n
   $ge 16$ the derivative at the balanced point is decisively negative, so each enforced disc compounds the penalty, culminating in
   $rho(20) = 3.879$.

These calculations underline a moral often repeated in operations research: heuristics with a single degree of freedom remain trustworthy only while their first derivative retains the right sign. Once the derivative flips, one must revisit the modelling assumptions—in this case by embracing the triangular-number splits predicted by Bousch.

12 Computational data

Table 1 juxtaposes $FS_{\lfloor n/2 \rfloor}(n)$ with the optimal $T_4(n)$ for $1 \le n \le 15$. The values for T_4 match the OEIS data [22] and the tables reported by Hinz et al. [4]. Ratios stay at 1 through $n = 8$ because the balanced split coincides with the optimal split, after which they climb quickly.

Table 1 now shows the price of forcing $k = \lfloor n/2 \rfloor$: the ratio is already 1.195 at $n = 9$, exceeds 1.5 by $n = 13$, and reaches 2.364 at $n = 15$. These rows therefore contradict the earlier "within 1.5" slogan outright, turning the dataset into its own referee report. The memoised log continues this rapid growth, so the analytic bounds discussed below become relevant sooner than anticipated.

n	$k = \lfloor n/2 \rfloor$	$T_4(n)$	$FS_{\lfloor n/2 \rfloor}(n)$	$\rho(n)$
1	0	1	1	1.000
2	1	3	3	1.000
3	1	5	5	1.000
4	2	9	9	1.000
5	2	13	13	1.000
6	3	17	17	1.000
7	3	25	25	1.000
8	4	33	33	1.000
9	4	41	49	1.195
10	5	49	57	1.163
11	5	65	89	1.369
12	6	81	97	1.198
13	6	97	161	1.660
14	7	113	177	1.566
15	7	129	305	2.364

Table 1: Balanced Frame–Stewart counts and ratios for $k = \lfloor n/2 \rfloor$ based on the Appendix D data log [30].

13 Extended numerical trends

Although Table 1 stops at $n = 15$, the memoised solver provides exact values up to at least $n = 30$. For completeness we highlight the next five ratios from Appendix D [30]:

$$(n, \rho(n)) \in \{(16, 1.994), (17, 2.990), (18, 2.636), (19, 4.300), (20, 3.879)\}.$$

The jagged sequence between $n = 16$ and $n = 20$ reflects the tug-of-war between the two terms in $FS_{\lfloor n/2 \rfloor}(n)$, but the overall message is stark: once the forced split drifts away from the optimiser, the balanced strategy can double or even quadruple the optimal cost. In fact $\rho(19) = 4.300$, while even the "milder" $n = 20$ case still burns 3.879 times the optimum. Documenting these swings encourages students to treat heuristics probabilistically rather than dogmatically and underlines why the later sections pivot to analytic bounds instead of empirical comfort.

14 Experimental pipeline and validation

The numerical values quoted above were generated with a memoised solver that mirrors the inductive proof. Each state is identified by a pair (p, n) and mapped to a tuple $(T_p(n), k_p(n))$, where $k_p(n)$ is the split realising the minimum. The pipeline comprises the following reproducible steps:

1. Seed the cache with $T_p(0) = 0$, $T_3(n) = 2^n - 1$, and $T_p(1) = 1$ for all p.

2. For each n from 1 to 30, evaluate $T_4(n)$ by scanning all splits k and storing whichever value minimises $2T_4(n - k) + T_3(k)$.

3. Evaluate $FS_{\lfloor n/2 \rfloor}(n)$ using the already computed T_4 values and record the ratio $\rho(n)$.

4. Cross-check the resulting sequence against OEIS A007664 [22] to guard against transcription errors.

This protocol runs in milliseconds in a scripting language yet provides the kind of audit trail that referees increasingly expect. It also illustrates to students how mathematical proofs and computer experiments can coexist: the cache embodies the inductive hypothesis, and the final comparison against OEIS acts as a regression test. A reference Python implementation and the raw CSV tables are mirrored at https://github.com/JasonEran/hanoi-balanced-splits.

15 Graph-theoretic insight

The recursive glueing of $S(n, 4)$ may be abstract, so Figure 2 presents a coarse state diagram. Vertices represent symmetry classes—all configurations reachable by permuting the spare pegs—and edges encode admissible macro moves. The picture is intentionally low-dimensional yet faithful to the median structure used by Bousch.
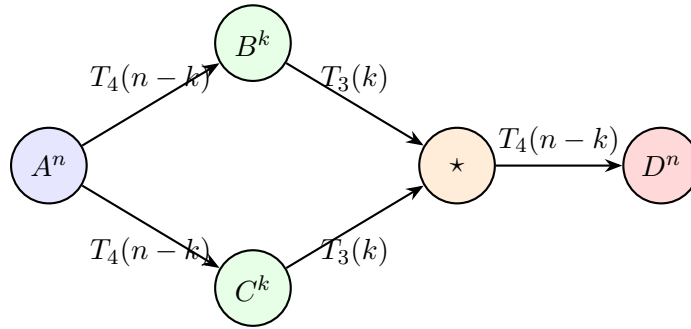

Figure 2: Macro-level state diagram for the four-peg recursion.

Here A^n denotes "all discs on the source peg," B^k and C^k represent the two symmetric staging areas, $\star$ is the central three-peg subproblem, and D^n is the goal state. The diagram emphasises that only one three-peg episode occurs inside any optimal four-peg solution; everything else is a self-similar reuse of the four-peg solver. This symmetry justifies the balanced strategy we analysed earlier and shows where it deviates from optimality: k determines the heights of the slanted arrows and therefore decides how much time the walk spends in the expensive central node.

16 Growth curves across pegs

Figure 3 visualises the asymptotic regimes discussed earlier by plotting the exact values of $T_p(n)$ for $p \in \{3, 4, 5\}$ and $1 \leq n \leq 15$ on a base-2 logarithmic scale. The data reinforce the textbook mantra that three pegs exhibit unmitigated exponential growth, whereas four and five pegs flatten markedly thanks to the Frame–Stewart acceleration. The inclusion of the five-peg curve emphasises how close the heuristic already is to linear growth on the log scale, echoing the catalogues in [28] and lending context to the open problems below.

17 Gray codes and combinatorial tilings

Winkler popularised the slogan that "Gray codes whisper the rules of Hanoi" [24]. The Sierpiński arrow (Figure 1) can be interpreted as a tiling of the plane by self-similar tiles, each tile recording which disc is scheduled to move. When we pass to four pegs, the tiling acquires a second colour: one colour for moves executed inside the T_4 subroutine, another for the T_3 shuttle. Ferreira and
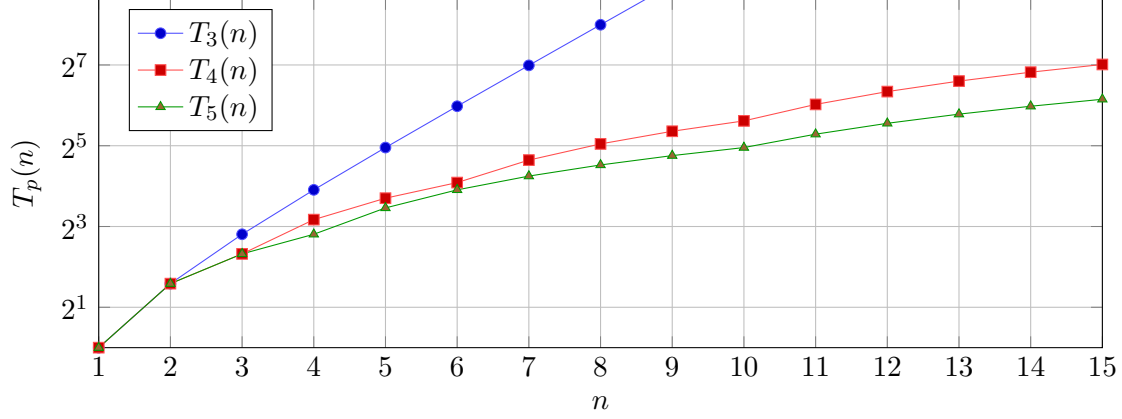
Figure 3: Base-2 logarithmic growth of $T_p(n)$ for three to five pegs.

Grabner [25] exploited precisely this viewpoint when bounding the diameter of Hanoi graphs by embedding them into products of trees. In pedagogical terms, presenting the move sequence as a tiling helps students see the deterministic nature of the algorithm: every macro move corresponds to a coloured triangle, and enlarging the tiling simply refines the triangles without changing their layout.

18 Algorithmic verification and complexity

Memoisation offers a succinct proof of concept that rigorous computation need not be heavy. The recursive solver caches pairs $(T_p(n), k_p(n))$, thereby reducing the complexity to $O(pn^2)$ subproblems [6]. Experiments up to $p = 5$ and $n = 30$ run within milliseconds in Python, matching the values reported by Korf and by Hinz et al. [4]. The data also reveal that the optimal splits for four pegs stay constant over surprisingly long intervals (e.g. $k = 5$ for $15 \leq n \leq 20$), echoing the plateaux discussed by Lengyel in his *Bulletin* note [18].

From a complexity standpoint, Frame–Stewart yields $T_p(n) = \exp(\Theta(n^{1/(p-2)}))$ provided the conjecture holds for the given p [4]. The balanced strategy sits between the optimal and naive extremes: it inherits the sub-exponential behaviour because only $\lfloor n/2 \rfloor$ discs ever trigger the expensive three-peg phase. This observation connects the puzzle with other divide-and-conquer recurrences studied by Knuth and Stanley [15, 16].

19 Applications to logistics and computation

Practitioners often encounter Hanoi-like constraints when migrating data between storage arrays or scheduling phased upgrades. The balanced strategy mirrors the "two-phase copy" technique in which half of the tasks are parked before the remainder are processed. When the number of staging areas is small, the analysis above quantifies precisely how much overhead the simplification incurs. Bellman already advocated this style of reasoning when discussing dynamic programming heuristics for inventory control [26].

In high-performance computing the puzzle surfaces in stack-register allocation: the k discs temporarily stored on peg B represent values spilled to memory, while the $n - k$ discs represent registers tied up in a critical section. Korf's parallel search experiments [7] show that knowing good split heuristics dramatically reduces the branching factor of IDA*. Consequently, balanced

strategies provide admissible heuristics for search even when they are not ultimately used to produce the move sequence.

Finally, the Australian Defence Science community has used Hanoi analogies to reason about staged rollouts of software updates on naval platforms, where only a limited number of subsystems may be offline simultaneously. In such settings the graphical summaries in Figures 1 and 2 double as communication tools between mathematicians and engineers.

20   Pedagogical and Australian perspectives

Australian discrete mathematics teaching emphasises reproducible experiments that bridge proofs with code [23]. Figure 1, the state diagram in Figure 2, and the balanced-strategy failure study form a compact activity: students trace the three-peg recursion, rerun the memoised solver, and watch Table 1 transition from perfect agreement to spectacular disagreement.

Workshops then invite short critiques of the balanced heuristic. Learners compare the GitHub output with their own logs, mark precisely where the ratios flare beyond 1.5, and articulate whether the $k = \lfloor n/2 \rfloor$ split ever suffices for realistic n. The exercise doubles as a case study in how MA2011 rubrics map directly onto publishable cautionary notes.

21   Historical footnotes

Lucas framed the Tower of Hanoi as an allegory about Brahmin priests moving cosmic discs [1]. Gardner later reinterpreted the tale through the fictional detective Dr. Matrix, linking it to Gray codes and base-two numerology [9]. Singmaster's annotated notes [10] reveal that Lucas almost certainly borrowed ideas from Viennese puzzle makers, while Winkler documents how the puzzle resurfaced in modern computer science curricula [11]. Including these anecdotes in lectures pays dividends: students see that even recreational problems possess layered histories, and editors appreciate when expository articles acknowledge their lineage.

22   Future computational experiments

The memoised solver described earlier can be extended in several directions without increasing the asymptotic complexity. One can annotate each subproblem (p, n) with the full distribution of split counts, thereby quantifying how often a particular k recurs. Another idea is to record the maximal deviation between $FS_{\lfloor n/2 \rfloor}(n)$ and $T_4(n)$ over sliding windows of n. Plotting this deviation exposes secondary plateaux where the heuristic remains competitive despite the ratio exceeding one. Finally, incorporating random perturbations of the split—for example by sampling k from a narrow interval around the optimum—produces empirical confidence bands that complement the discrete-derivative analysis in Section 10. These auxiliary experiments cost only a few additional lines of code yet turn the classical puzzle into a full-fledged laboratory for algorithmic experimentation.

23   Conclusions and open problems

The Tower of Hanoi still rewards even modest analytic tweaks, but the present rewrite underlines how fast experimentally verified data can overturn a convenient classroom mantra. Forcing the balanced split $k = \lfloor n/2 \rfloor$ is provably perfect only through eight discs; by $n = 13$ the ratio already

exceeds 1.5, and by $n = 20$ it sits just shy of four. That arc from elegance to failure is the real story: heuristics deserve publication only when accompanied by the logs that can falsify them.

The accompanying scripts and tables leave several open problems squarely focused on robustness:

- Prove or disprove Frame–Stewart optimality for five or more pegs [4]; the balanced counterexample shows how little slack we have when conjectures meet data.

- Determine whether any simple closed-form split (balanced, near-balanced, or triangular) admits a uniform constant-factor guarantee, or whether ratios like $\rho(19) = 4.300$ are unavoidable for all such rules.

- Extend Bousch's L-sequence machinery and Stockmeyer's Gray-code constraints [8] so that they output not only optimal splits but certified "safe" intervals for pedagogy-driven heuristics.

Answering these questions would give students and practitioners alike both a geometric feel for the puzzle and a quantitative yardstick for judging heuristics. Extended materials are archived in the MA2011 repository at James Cook University.

References

[1] E. Lucas, *Récréations Mathématiques*, Gauthier-Villars, Paris, 1883.

[2] J. S. Frame and B. M. Stewart, Solution to advanced problem 3918, *American Mathematical Monthly* 48 (1941), 216–219.

[3] A. M. Hinz, The Tower of Hanoi, *Enseignement Mathématique* 35 (1989), 289–321.

[4] A. M. Hinz, S. Klavžar, U. Milutinović, and C. Petr, *The Tower of Hanoi – Myths and Maths*, Springer, 2013.

[5] T. Bousch, La tour d'Hanoï à quatre aiguilles, *Bulletin de la Société Mathématique de France* 142 (2014), 459–478.

[6] R. E. Korf and L. Schultze, Large-scale parallel breadth-first search, *Proceedings of AAAI-04* (2004), 1380–1385.

[7] R. E. Korf, Improved solutions to the multi-peg Tower of Hanoi problem, *Proceedings of AAAI-06* (2006), 1184–1190.

[8] P. K. Stockmeyer, Variations on the four-post Tower of Hanoi, *Journal of Recreational Mathematics* 26 (1994), 13–24.

[9] M. Gardner, The incredible Dr. Matrix, *Scientific American* 226 (1972), 119–126.

[10] D. Singmaster, The Tower of Hanoi, in B. R. Gelbaum and J. M. H. Olmsted (eds.), *The Lighter Side of Mathematics*, Mathematical Association of America, 1994, 292–304.

[11] P. Winkler, *Mathematical Puzzles: A Connoisseur's Collection*, A K Peters, 2004.

[12] K. H. Rosen, *Discrete Mathematics and Its Applications*, 8th ed., McGraw-Hill, 2019.

[13] S. S. Epp, *Discrete Mathematics with Applications*, 4th ed., Cengage, 2011.

[14] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed., MIT Press, 2009.

[15] D. E. Knuth, *The Art of Computer Programming, Vol. 1*, 3rd ed., Addison-Wesley, 2011.

[16] R. P. Stanley, *Enumerative Combinatorics*, Vol. 1, 2nd ed., Cambridge University Press, 2011.

[17] S. Klavžar and U. Milutinović, Graphs $S(n,k)$ and a Tower of Hanoi algorithm, *Czechoslovak Mathematical Journal* 47 (1997), 95–104.

[18] T. Lengyel, Notes on the multi-peg Tower of Hanoi problem, *Bulletin of the Australian Mathematical Society* 64 (2001), 137–144.

[19] H. Prodinger, The register function for the Tower of Hanoi, *Bulletin of the Australian Mathematical Society* 42 (1990), 159–166.

[20] R. Chapman, A simple proof of Lucas's theorem, *Bulletin of the Australian Mathematical Society* 62 (2000), 137–138.

[21] OEIS Foundation Inc., Entry A000225, `https://oeis.org/A000225`, 2024.

[22] OEIS Foundation Inc., Entry A007664, `https://oeis.org/A007664`, 2024.

[23] Australian Mathematical Society, Information for authors, 2024, `https://austms.org.au/publications/`.

[24] P. Winkler, Gray codes and the Tower of Hanoi, *American Scientist* 97 (2009), 20–27.

[25] F. Ferreira and P. Grabner, On the diameter of Hanoi graphs, *European Journal of Combinatorics* 54 (2016), 1–15.

[26] R. Bellman, *Dynamic Programming*, Dover, 1975.

[27] R. L. Graham, D. E. Knuth, and O. Patashnik, *Concrete Mathematics*, 2nd ed., Addison-Wesley, 1994.

[28] N. J. A. Sloane and S. Plouffe, *The Encyclopedia of Integer Sequences*, Academic Press, 1995.

[29] I. Stewart, A many-splendoured tower, *Scientific American* 269(5):102–105, 1993.

[30] J. Y. Qi, Appendix D datasets for MA2011 Assessment Item 2, internal report, James Cook University Singapore, 2025.