

On an Infinite Family of Solvable Hanoi Graphs

DANY AZRIEL, NOAM SOLOMON, AND SHAY SOLOMON

Ben-Gurion University of the Negev

Abstract. The Tower of Hanoi problem is generalized by placing pegs on the vertices of a given directed graph G with two distinguished vertices, S and D , and allowing moves only along arcs of this graph. An optimal solution for such a graph G is an algorithm that completes the task of moving a tower of any given number of disks from S to D in a minimal number of disk moves.

In this article we present an algorithm which solves the problem for two infinite families of graphs, and prove its optimality. To the best of our knowledge, this is the first optimality proof for an *infinite* family of graphs.

Furthermore, we present a unified algorithm that solves the problem for a wider family of graphs and conjecture its optimality.

Categories and Subject Descriptors: F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems; G.2.1 [Discrete Mathematics]: Combinatorics—*Combinatorial algorithms*

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Optimality proofs, Tower of Hanoi

ACM Reference Format:

Azriel, D., Solomon, N., and Solomon, S. 2008. On an infinite family of solvable Hanoi graphs. *ACM Trans. Algor.* 5, 1, Article 13 (November 2008), 22 pages. DOI = 10.1145/1435375.1435388 <http://doi.acm.org/10.1145/1435375.1435388>

1. Introduction

The Tower of Hanoi puzzle, invented in 1883 by Lucas [1893], consists of three distinct pegs S , A , and D , and a set of n pierced disks of differing diameter that can be stacked on the pegs. Initially, all disks are stacked on one peg (the source, S) in decreasing order of size, from bottom to top. The task is to move the n disks to some other peg (the destination, D), using A as an auxiliary peg, subject to the following two rules.

This research has been Partially supported by the Lynn and William Frankel Center for Computer Science.

Authors' addresses: D. Azriel, N. Solomon, Department of Mathematics, Ben-Gurion University of the Negev, Israel, e-mail: {azrield,noamso}@math.bgu.ac.il. S. Solomon, Department of Computer Science, Ben-Gurion University of the Negev, Israel, e-mail: shayso@cs.bgu.ac.il.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or direct commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or permissions@acm.org.
© 2008 ACM 1549-6325/2008/11-ART13 \$5.00 DOI 10.1145/1435375.1435388 <http://doi.acm.org/10.1145/1435375.1435388>

- (R1) Each move consists of taking the top-most disk from a peg and placing it on top of all disks residing on some other peg. (Thus, pegs behave as stacks.)
- (R2) At no time may a disk be placed upon a smaller one.

It is well known that the problem can be solved for any $n \geq 1$; furthermore, it can be verified that $2^n - 1$ moves are necessary and sufficient to carry out this task.

There has been continuing interest in the problem from several points of view. For educational purposes, it serves in introductory programming courses as a very good example of a recursive program. In fact, as opposed to many other recursive procedures, here it is much more elaborate to present an iterative solution. Various properties of instances of the problem were studied. In Hinz [1992], analogies between Pascal's triangle, the Sierpiński gasket, and the Tower of Hanoi are found. Properties of the solutions are also discussed, as in Allouche et al. [1994], where it is shown that, with direct approach coding, a string that represents an optimal solution is square-free.

The problem calls for generalizations. Various generalizations, such as having any initial and final configurations [Er 1985] and assigning colors to disks [Minsker 1989], were studied. A common generalization of the problem is to allow more than three pegs and put restrictions on the legitimate moves of the disks. This is discussed in Stewart [1941, 1939], Frame [1941], Leiss [1984, 1983], and Azriel and Berend [2006]. Several cases of relaxation of rule (R2), where a disk may be placed on top of a smaller disk under some conditions, are discussed in Wood [1981], Poole [1992], and Dinitz and Solomon [2008, 2006]. The case of four pegs instead of three and the correspondence with graphs is discussed thoroughly in Stockmeyer [1994].

We are interested in the following generalization of the problem: A *Hanoi graph* is a simple, directed, finite graph $G = (V, E)$ with two distinguished vertices, denoted by S and D , $S \neq D$, such that (without loss of generality) for each vertex $v \in V$ there is a path from S to v and a path from v to D . At each vertex of G , there is a peg which we identify with the vertex itself. The source initially contains m disks of different sizes, such that smaller disks rest on top of larger ones. The task is to move all disks from S to D using the other vertices of the graph as auxiliary pegs. The transfer is subject to rules (R1) and (R2), and another rule is added.

- (R3) A disk may be moved from a peg v to another peg w only if there is an arc from v to w , namely, $(v, w) \in E$.

The Tower of Hanoi problem $HAN(G, m)$ for a Hanoi graph G and $m \geq 0$ is to transfer m disks of distinct sizes from S to D , subject to rules (R1)–(R3).

A problem $HAN(G, m)$ is *solvable* if the aforementioned task can be accomplished. A Hanoi graph G is a *solvable graph* if $HAN(G, m)$ is such for all $m \geq 1$.

The requirement that for each vertex $v \in V$, there is a path from S to v and a path from v to D is designed to get rid of inessential vertices. In fact, if there is no path from S to v or from v to D , then no solution to $HAN(G, m)$ may use the peg v , so that v may be omitted from the graph. We say that a Hanoi graph is *reduced* if it does not contain any inessential vertices.

Leiss [1983] obtained the following characterization of solvable Hanoi graphs. (Note that in his formulation of the problem, V may contain inessential vertices. Hence the formulation of the theorem hereby given is modified accordingly.)

THEOREM A [LEISS 1983]. *Let $G = (V, E)$ be a reduced Hanoi graph, and let $G^* = (V, E^*)$ be its transitive closure: $E^* = \{(v, w) : v \neq w, \text{ there is a path from } v \text{ to } w\}$. Then G is solvable if and only if G^* contains a clique¹ of size 3.*

For a solvable Hanoi graph G , an optimal solution to the problem $HAN(G, n)$ is an algorithm that completes the task of moving a tower of n disks from S to D in a minimal number of disk moves. Optimal solutions for the five nonisomorphic solvable Hanoi graphs on three vertices are discussed thoroughly in Sapir [2004]. The task of finding optimal solutions for various solvable Hanoi graphs on four vertices may already be very difficult. For example, the problem $HAN(K_4, n)$, where K_4 is the complete graph on four vertices, first appeared in Dudeney [1907]. Frame [1941] and Stewart [1941] introduced essentially the same algorithm scheme for the problem. It has been verified that, up to 30 disks, it does give the minimal number of moves [Korf and Felner 2007]. In Szegedy [1999] a lower bound for the number of moves is established, and in Chen and Shen [2004] this bound is significantly improved. However, a justification for the optimality of that algorithm is still lacking today. Knuth is reported to have said “I doubt if anyone will ever resolve this conjecture; it is truly difficult” [Lunnon 1986]. It might be hoped that in simpler cases like the cyclic and “k-in-a-row” graphs, the question of finding an optimal solution would be easier, as there is less freedom at each move. However, optimality proofs for these “simpler” cases, as in the case of Frame-Stewart, seems currently out of reach [Scorer et al. 1944; Stockmeyer 1994].

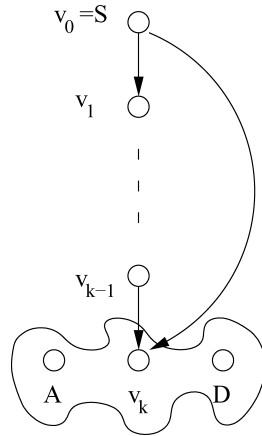
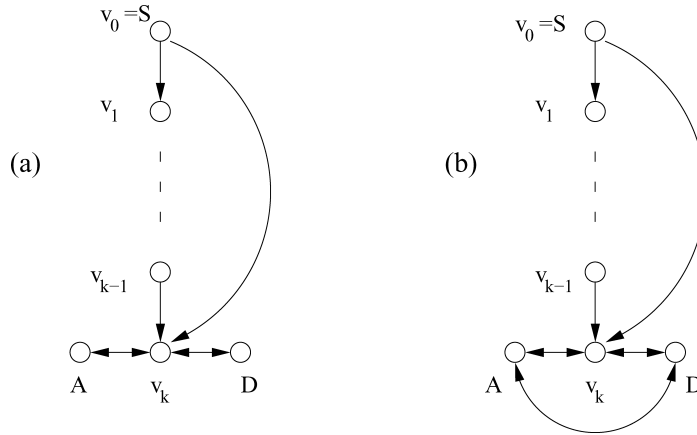
We believe it is interesting to explore an infinite family of solvable Hanoi graphs for which we can find an optimal solution. Denote by Γ the family of Hanoi graphs with exactly one strongly connected component of size 3 (i.e., any other strongly connected component must be of size at most 2).

In this article we study the family $\Gamma' \subset \Gamma$ of Hanoi graphs on at least 3 vertices, denoted by $\{A, D, S = v_0, \dots, v_k\}$, $k \geq 0$, with exactly one strongly connected component on 3 vertices, A , D , and v_k , a directed path $P_k = (S = v_0, \dots, v_k)$ of length k from S to v_k , an arc (S, v_k) , and no other arcs, as depicted in Figure 1. Observe that Γ' can be partitioned into 18 subsets according to the strongly connected component of size 3. Each such subset of Γ' is different from the other 17 subsets only in arcs between the 3 vertices A , D , and v_k of the strongly connected component; denote by Γ'_G such a subset, where G is some strongly connected graph on A , D , and v_k . Another natural partition of Γ' can be made according to the length k of the directed path P_k ; denote by Γ'_k such a subset, $k \geq 0$. Denote by G_k the unique graph in $\Gamma'_k \cap \Gamma'_G$. We denote by R the “3-in-a-row” graph on the vertices A , v_k , and D , in this order, and by C the complete graph on these vertices.

Our main focus is on the two infinite families of Hanoi graphs, Γ'_R and Γ'_C , which are symmetric in a sense. (Observe that $\Gamma'_R = \{R_k : k \geq 0\}$ and $\Gamma'_C = \{C_k : k \geq 0\}$.) The general member of the former (respectively, latter) family is the Hanoi graph depicted in Figure 2(a) (respectively, Figure 2(b)).

The article is organized as follows. In Sections 3 and 4 we present an algorithm that solves both $HAN(R_k, n)$ and $HAN(C_k, n)$, for all $k \geq 0$, and prove its optimality. For $k = 0$, our algorithm coincides with Sapir’s [Sapir 2004] unified algorithm that

¹A clique in a directed graph is a subset of vertices in the graph such that for every two vertices in the clique, there exists a bidirectional arc connecting the two.

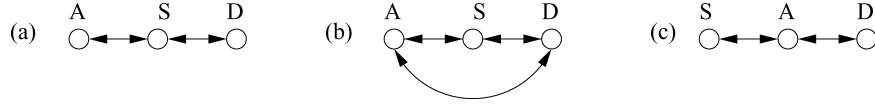
FIG. 1. A general member in Γ'_k .FIG. 2. An illustration of: (a) R_k ; and (b) C_k .

solves $\Gamma'|_0$. For $k = 1, 2$, our algorithm is optimal for all values of n , as in $k = 0$, whereas for $k \geq 3$, our algorithm is optimal only for $n \geq 2k - 2$ (see Section 6 for further details). In Section 5 we present a unified algorithm that solves Γ'_k , for all $k \geq 1$. We prove its optimality for $k = 1, 2$, and conjecture its optimality for all $k \geq 3$.

2. Definitions and Notation

In this article, a disk set \mathcal{D} is a subset of $\{1, \dots, n\}$. A disk set of size m refers to $\{1, \dots, m\}$, unless otherwise mentioned. A *configuration* of a disk set \mathcal{D} is a specification of disjoint ordered sets of disks on the pegs whose union is \mathcal{D} . A move of disk r from peg u to peg v is denoted by $u \xrightarrow{r} v$.

Obviously, a configuration of $\mathcal{D} \setminus \{r\}$ is the same before and after such a move; we refer to it as the *configuration of $\mathcal{D} \setminus \{r\}$ during $u \xrightarrow{r} v$* . (Even though a move

FIG. 3. An illustration of: (a) R_0 ; (b) C_0 ; and (c) Aux_0 .

is considered an atomic operation, we view the word “during” as most intuitively appropriate.)

The *length* of a move sequence P is the number of moves in it, denoted by $|P|$. Given a move sequence P , containing at some point the move $u \xrightarrow{r} v$ and at some later point the move $u' \xrightarrow{r'} v'$, the subsequence of P of all moves between these moves, including the former move but not the latter, is denoted by $P[u \xrightarrow{r} v, u' \xrightarrow{r'} v']$, or simply $[u \xrightarrow{r} v, u' \xrightarrow{r'} v']$ if the move sequence is clear from the context. (If either of the moves is performed more than once, we shall specify which occurrence of that move is considered.) Similarly, $P(u \xrightarrow{r} v, u' \xrightarrow{r'} v')$ denotes the subsequence of P between these moves, excluding the former and latter moves. In general, we may also use open-closed and closed-closed intervals.

For a move sequence P , denote the first (respectively, last) move of disk i in P by $First(i)$ (respectively, $Last(i)$), for any i in \mathcal{D} .

For any configuration \mathcal{C} and disk set \mathcal{D} , we define the *restriction of \mathcal{C} to \mathcal{D}* , denoted by $\mathcal{C}|_{\mathcal{D}}$, as the configuration \mathcal{C} with all disks not in \mathcal{D} removed. Similarly, for any move sequence P of \mathcal{D} from a configuration \mathcal{C}_1 to another configuration \mathcal{C}_2 , and any disk subset \mathcal{D}' , we define the *restriction of P to \mathcal{D}'* , denoted by $P|_{\mathcal{D}'}$, as the result of omitting all moves of disks not in \mathcal{D}' from P ; clearly, $P|_{\mathcal{D}'}$ transforms $\mathcal{C}_1|_{\mathcal{D}'}$ to $\mathcal{C}_2|_{\mathcal{D}'}$. It is easy to see that the restriction of any legal move sequence to any disk subset is legal as well. Thus, for any two disk sets $\mathcal{D}' \subset \mathcal{D}$ and any move sequence P of \mathcal{D} , the restriction $P|_{\mathcal{D}'}$ is a legal move sequence of \mathcal{D}' . Moreover, the restriction $P|_{\mathcal{D} \setminus \mathcal{D}'}$ is a legal move sequence of $\mathcal{D} \setminus \mathcal{D}'$, and $|P|_{\mathcal{D} \setminus \mathcal{D}'} + |P|_{\mathcal{D}'} = |P|$ holds.

We use the notation H_k to denote both R_k and C_k , when it is clear from the context to which we refer. This notation enables us to achieve uniformity for the two aforementioned families of graphs $\Gamma'_R = \{R_k : k \geq 0\}$ and $\Gamma'_C = \{C_k : k \geq 0\}$.

In what follows, we do not distinguish between a move sequence and an algorithm generating it, if this does not lead to a misunderstanding.

3. The Infinite Family Γ'_R

In this section we define $H_k = R_k$.

3.1. MAIN RESULTS. Denote the Hanoi graph depicted in Figure 3(c) by Aux_0 . The following algorithm $MoveAux_0(n) = MoveAux_0(n, S, A, D)$ solves $HAN(Aux_0, n)$. The input of this algorithm consists of the number n of disks to be moved, the source peg S , the destination peg D , and an auxiliary peg A . Even though Aux_0 is not a member of the family $\{H_k : k \geq 0\}$, it plays an important role, as $MoveAux_0(n)$ is embedded in all the algorithms presented in the sequel.

Algorithm 1. *MoveAux₀(n, S, A, D)*:

The algorithm moves n disks of distinct sizes on the graph Aux_0 . The disks are initially stacked on S and are moved to D , using A as auxiliary.

```

1: if  $n > 0$  then
2:    $MoveAux_0(n - 1, S, A, D)$ 
3:    $S \xrightarrow{n} A$ 
4:    $MoveAux_0(n - 1, D, A, S)$ 
5:    $A \xrightarrow{n} D$ 
6:    $MoveAux_0(n - 1, S, A, D)$ 
7: end if

```

Algorithm 2. *MoveH₀(n, S, A, D)*:

The algorithm moves n disks of distinct sizes on the graph H_0 . The disks are initially stacked on S and are moved to D , using A as auxiliary.

```

1: if  $n > 0$  then
2:    $MoveH_0(n - 1, S, D, A)$ 
3:    $S \xrightarrow{n} D$ 
4:    $MoveAux_0(n - 1, A, S, D)$ 
5: end if

```

Algorithm 3. *MoveH₁(n, S, A, D)*:

The algorithm moves n disks of distinct sizes on the graph H_1 . The disks are initially stacked on S and are moved to D , using v_1 and A as auxiliary pegs.

```

1: if  $n > 0$  then
2:    $MoveH_1(n - 1, S, D, A)$ 
3:    $S \xrightarrow{n} v_1; v_1 \xrightarrow{n} D$ 
4:    $MoveAux_0(n - 1, A, v_1, D)$ 
5: end if

```

Denote the Hanoi graph depicted in Figure 3(a) by H_0 . The algorithm $MoveH_0(n) = MoveH_0(n, S, A, D)$ solves $HAN(H_0, n)$.

Algorithms $MoveAux_0(n)$ and $MoveH_0(n)$ are in fact particular instances of Sapir's [Sapir 2004] unified algorithm, which provides an optimal solution to the problem of moving a pile of n disks between any two vertices of a strongly connected graph on 3 vertices [Sapir 2004]. They are easily shown to require $3^n - 1$ and $\frac{3^n - 1}{2}$ moves, respectively.

Consider the graph H_1 . The algorithm $MoveH_1(n) = MoveH_1(n, S, A, D)$ solves $HAN(H_1, n)$.

Observe that the move sequence $MoveH_1(n)$ is obtained from $MoveH_0(n)$ by replacing the *first* move $S \xrightarrow{i} C$ of disk i , where C is either A or D , with the two following moves: $S \xrightarrow{i} v_1, v_1 \xrightarrow{i} C$, for all $1 \leq i \leq n$.

THEOREM 3.1. *An optimal solution to $HAN(H_1, n)$ is provided by $MoveH_1(n)$, and it takes $\frac{3^n + 2n - 1}{2}$ moves.*

Algorithm 4. $MoveH_2(n, S, A, D)$:

The algorithm moves n disks of distinct sizes on the graph H_2 . The disks are initially stacked on S and are moved to D , using v_1, v_2 and A as auxiliary pegs.

```

1: if  $n = 1$  then
2:    $S \xrightarrow{n} v_2; v_2 \xrightarrow{n} D$ 
3: else if  $n \geq 2$  then
4:    $MoveH_2(n - 2, S, D, A)$ 
5:    $S \xrightarrow{n-1} v_1$ 
6:    $S \xrightarrow{n} v_2; v_2 \xrightarrow{n} D$ 
7:    $v_1 \xrightarrow{n-1} v_2; v_2 \xrightarrow{n-1} D$ 
8:    $MoveAux_0(n - 2, A, v_2, D)$ 
9: end if

```

Consider the graph H_2 . The algorithm $MoveH_2(n) = MoveH_2(n, S, A, D)$ solves $HAN(H_2, n)$.

THEOREM 3.2. *An optimal solution to $HAN(H_2, n)$ is provided by $MoveH_2(n)$, and it takes $\frac{3^n + (-1)^n + 16n - 2}{8}$ moves.*

Next, we present an algorithm that solves $HAN(H_k, n)$, for a general $k \geq 1$, and declare its optimality. We denote this algorithm by $MoveH_k(n) = MoveH_k(n, S, A, D)$. For the case $k \geq 3$, we assume that $n \geq 2k - 2$. Let $MoveH_1^k(n)$ be the move sequence obtained from $MoveH_1(n)$ by replacing v_1 with v_k in all disk moves. It is easy to see that $MoveH_1^k(n)$ is a *feasible* solution to $HAN(H_k, n)$, for all $k \geq 1$.

Let us analyze $MoveH_k(n)$ for $k \geq 3, n \geq 2k - 2$. Lines 6–12 transfer disks $1, \dots, k - 1$ from S to pegs v_{k-1}, \dots, v_1 , respectively. Line 13 transfers disks k, \dots, n from S to D . Lines 14–28 transfer disks $1, \dots, k - 1$ from pegs v_{k-1}, \dots, v_1 to D , respectively. Note that lines 14–28 can be equivalently replaced by the move sequence $MoveP_k(k - 1, v_k, A, D)$, obtained from $MoveH_0(k - 1, v_k, A, D)$ by replacing the first move $v_k \xrightarrow{i} C$ of disk i , where C is either A or D , with the move sequence that carries i along the path (v_{k-i}, \dots, v_k, C) , for all $1 \leq i \leq k - 1$.

The following theorem is the main result of this article, and includes Theorems 3.1 and 3.2 as particular instances.

THEOREM 3.3. *An optimal solution to $HAN(H_k, n)$ is provided by $MoveH_k(n)$ for any $k \geq 1$ and $n \geq 2k - 2$. For $k \leq 2$, it is optimal for all values of n . The number of moves it takes is*

$$\begin{cases} \frac{3^n + (-1)^n + 16n - 2}{8}, & k = 2, \\ \frac{3^{n-k+1} + 2n + 3^{k-1} + 2(k-1)^2 - 2}{2}, & k = 1, k \geq 3. \end{cases}$$

3.2. AUXILIARY STATEMENTS. Denote the minimum number of moves needed to solve $HAN(H_k, n)$ (respectively, $HAN(Aux_0, n)$) by $h_n^{(k)}$ (respectively, $h_n^{(aux_0)}$).

Fact 3.4. For all $k \geq 0$ and $n \geq 1$, $h_n^{(k)} > h_{n-1}^{(k)}$.

Algorithm 5. $MoveH_k(n, S, A, D)$:

The algorithm moves n disks of distinct sizes on the graph H_k . The disks are initially stacked on S and are moved to D , using v_1, \dots, v_k and A as auxiliary pegs.

```

1: if  $k = 1$  then
2:    $MoveH_1(n, S, A, D)$ 
3: else if  $k = 2$  then
4:    $MoveH_2(n, S, A, D)$ 
5: else
6:   if  $n \geq 2k - 2$  then
7:      $v_0 = S$ 
8:     for  $j = 1$  to  $k - 1$  do
9:       for  $l = 0$  to  $k - j - 1$  do
10:         $v_l \xrightarrow{j} v_{l+1}$ 
11:       end for
12:     end for
13:      $MoveH_k^k(n - k + 1, S, A, D)$ 
14:     for  $j = 1$  to  $k - 1$  do
15:       for  $l = k - j$  to  $k - 1$  do
16:         $v_l \xrightarrow{j} v_{l+1}$ 
17:       end for
18:       if  $j \equiv k - 1 \pmod{2}$  then
19:          $v_k \xrightarrow{j} D$ 
20:          $B = A$ 
21:          $C = D$ 
22:       else
23:          $v_k \xrightarrow{j} A$ 
24:          $B = D$ 
25:          $C = A$ 
26:       end if
27:        $MoveAux_0(j - 1, B, v_k, C)$ 
28:     end for
29:   end if
30: end if

```

The following lemma holds for all graphs in this family. We use it to prove Theorem 3.3.

LEMMA 3.5. *For all $n \geq 1$ and $k \geq 1$, in any optimal solution to $HAN(H_k, n)$ the largest disk is moved just twice: $S \xrightarrow{n} v_k$ and $v_k \xrightarrow{n} D$. Furthermore, there exists an optimal solution in which no other disk moves are performed between these two moves.*

PROOF. Let SOL be an optimal solution to $HAN(H_k, n)$. The last move of disk n must be $v_k \xrightarrow{n} D$. Omit all previous moves of disk n , and replace that last move with the two following moves: $S \xrightarrow{n} v_k, v_k \xrightarrow{n} D$. Clearly the resulting sequence is also a solution to $HAN(H_k, n)$, in which the largest disk is moved just twice, and between these two moves no other disk moves are performed. In SOL , disk n moves

just twice, otherwise, the resulting sequence is shorter than *SOL*, contradicting the assumption that *SOL* is optimal. \square

The following simple observation is used in the sequel.

Observation 3.6. Let *SOL* be a solution to $HAN(H_k, n)$, with $k \geq 2$. In any configuration reached during *SOL*, the following holds.

—The pegs v_1, \dots, v_{k-1} contain at most one disk each.

—If l_i resides on v_i and l_j resides on v_j , $1 \leq i < j \leq k - 1$, then $l_i > l_j$.

3.3. PROOF OF THEOREM 3.1. During the last move $v_1 \xrightarrow{n} D$ of disk n , all $n - 1$ smaller disks reside on peg A . Hence, before disk n makes its last move, at least $h_{n-1}^{(1)}$ moves of the $n - 1$ smaller disks are made. At some point, all $n - 1$ smaller disks are gathered on peg D . The minimum number of moves needed to transfer these disks from A to D is at least $h_{n-1}^{(aux_0)} = 3^{n-1} - 1$. We get that $h_n^{(1)} \geq h_{n-1}^{(1)} + 2 + 3^{n-1} - 1$. The equation $|MoveH_1(n)| = |MoveH_1(n-1)| + 2 + 3^{n-1} - 1$ implies that $h_n^{(1)} \geq |MoveH_1(n)|$. The minimality of $h_n^{(1)}$ yields the opposite inequality. Therefore,

$$|MoveH_1(n)| = h_n^{(1)} = \frac{3^n + 2n - 1}{2},$$

and the theorem follows. \square

3.4. PROOF OF THEOREM 3.2. The proof in the case $n = 1$ is immediate.

For $n > 1$, we distinguish the two following cases.

- (1) *During the last move $v_2 \xrightarrow{n} D$ of disk n , peg v_1 is empty.* During this move, all $n - 1$ smaller disks reside on peg A . Hence, before this move at least $h_{n-1}^{(2)}$ moves of the $n - 1$ smaller disks are made. At some point, all $n - 1$ smaller disks are gathered on peg D . The minimum number of moves needed to transfer these disks from A to D is at least $h_{n-1}^{(aux_0)} = 3^{n-1} - 1$. Disk n must make at least two moves. Altogether, at least $h_{n-1}^{(2)} + 1 + 3^{n-1}$ moves are made.
- (2) *During the last move $v_2 \xrightarrow{n} D$ of disk n , peg v_1 is nonempty.* During this move, a single disk resides on peg v_1 , denoted by i , and $n - 2$ disks are gathered on peg A . Hence, before this move at least $h_{n-2}^{(2)}$ moves of disks in $\{1, \dots, n - 1\} \setminus \{i\}$ are made. At the final configuration, all disks in $\{1, \dots, n - 1\} \setminus \{i\}$ are gathered on peg D . The minimum number of moves needed to transfer these $n - 2$ disks from A to D is $h_{n-2}^{(aux_0)} = 3^{n-2} - 1$. Disk i must make at least three moves. Disk n must make at least two moves. Altogether, at least $h_{n-2}^{(2)} + 3^{n-2} + 4$ moves are made.

By Fact 3.4, for $n > 1$, $h_{n-2}^{(2)} + 3^{n-2} + 4 \leq h_{n-1}^{(2)} + 1 + 3^{n-1}$. Hence for $n > 1$, $h_n^{(2)} \geq h_{n-2}^{(2)} + 3^{n-2} + 4$. The equation $|MoveH_2(n)| = |MoveH_2(n-2)| + 3^{n-2} + 4$ implies that $h_n^{(2)} \geq |MoveH_2(n)|$. The minimality of $h_n^{(2)}$ yields the opposite inequality. Therefore,

$$|MoveH_2(n)| = h_n^{(2)} = \frac{3^n + (-1)^n + 16n - 2}{8},$$

and the theorem follows. \square

3.5. PROOF OF THEOREM 3.3. The proof of Theorem 3.3 for $k \leq 2$ is a direct corollary of Theorems 3.1 and 3.2. We henceforth restrict our attention to the case $k \geq 3, n \geq 2k - 2$.

LEMMA 3.7. *Let $n \geq 2k - 2, k \geq 3$. Then Algorithm $MoveH_k(n)$ solves $HAN(H_k, n)$ in $\frac{3^{n-k+1} + 2n + 3^{k-1} + 2(k-1)^2 - 2}{2}$ moves.*

PROOF. Let us analyze the number of moves required by Algorithm $MoveH_k(n)$ in the case $n \geq 2k - 2, k \geq 3$. The number of moves performed in the for-loop of lines 8–12 is $\sum_{j=1}^{k-1} j = \frac{k(k-1)}{2}$. By Theorem 3.1, the number of moves performed in line 13 of the algorithm is $\frac{3^{n-k+1} + 2(n-k+1) - 1}{2}$. The total number of moves in the for-loop of lines 14–28 is $\sum_{j=1}^{k-1} (3^{j-1} + j) = \frac{3^{k-1} - 1 + k(k-1)}{2}$. To see this, note that at the j th iteration of this loop, $1 \leq j \leq k - 1$, the number of moves performed until line 26 is $j + 1$, and $3^{j-1} - 1$ additional moves are performed in line 27. Altogether, the number of moves required by the algorithm is

$$\frac{3^{n-k+1} + 2n + 3^{k-1} + 2(k-1)^2 - 2}{2}. \quad \square$$

Definition 3.8. A configuration is called p -occupied if exactly p of the pegs v_1, \dots, v_{k-1} are occupied (with one disk each), $0 \leq p \leq k - 1$.

LEMMA 3.9. *Let \mathcal{C}_1 be a p -occupied configuration, with disks l_1, \dots, l_p residing on p of the pegs v_1, \dots, v_{k-1} . Let \mathcal{C}_2 be a configuration in which all these disks reside on peg C , where C is either A or D . The minimum number of moves made by disks l_1, \dots, l_p between configurations \mathcal{C}_1 and \mathcal{C}_2 , disregarding moves to pegs v_1, \dots, v_{k-1} , is at least $h_p^{(1)}$.*

PROOF. First, we show a reduction from configuration \mathcal{C}_1 to another configuration \mathcal{C}'_1 , obtained from it by relocating the p disks l_1, \dots, l_p to a single peg, v_{k-1} . Since moves of disks l_1, \dots, l_p to pegs v_1, \dots, v_{k-1} are disregarded, the first considered move of any such disk is from v_{k-1} to v_k . Observe that for all $x, y \in \{l_1, \dots, l_p\}$, such that $x < y$, disk x is moved from v_{k-1} before disk y is moved from it. Consequently, the configuration \mathcal{C}_1 in which the p disks l_1, \dots, l_p are gathered on p of the pegs v_1, \dots, v_{k-1} can be equivalently replaced by the configuration \mathcal{C}'_1 , where all these disks are gathered on peg v_{k-1} , from largest at the bottom to smallest at the top.

To conclude the proof, observe that the minimum number of moves made by disks l_1, \dots, l_p between configurations \mathcal{C}'_1 and \mathcal{C}_2 is no less than the minimum number of moves needed to solve $HAN(H_1, p)$, that is, $h_p^{(1)}$. \square

LEMMA 3.10. *There exists an optimal solution to $HAN(H_k, n)$, such that for each $1 \leq d \leq n$, during the first move of disk d , either all disks smaller than d (except those residing on pegs v_1, \dots, v_{k-1} at that time) reside on peg A or they all reside on peg D .*

PROOF. Assume that SOL is an optimal solution to $HAN(H_k, n)$. By using reverse induction (i.e., an induction that uses a negative in the inductive step), we build from SOL a solution that satisfies the conditions of the lemma. For each $1 \leq d \leq n$, denote by V_d the set of disks residing on pegs v_1, \dots, v_{k-1} during the first move of disk d .

- (1) By Lemma 3.5, there exists an optimal solution such that during the first move $S \xrightarrow{n} v_k$ of disk n , all disks smaller than n (except V_n) reside on peg A .
- (2) Take some $1 \leq d < n$ and assume that in *SOL*, during the first move $First(d+1)$ of disk $d+1$, either all disks smaller than $d+1$ (except V_{d+1}) reside on peg A or they all reside on peg D ; denote this peg by C .
 - (a) Assume that during the first move of disk $d+1$, disk d resides on one of the pegs v_1, \dots, v_{k-1} . Postpone all moves of disk d in the interval $[First(d), First(d+1))$ and perform them just before $First(d+1)$.
 - (b) Assume that during the first move of disk $d+1$, disk d resides on peg C . Notice that during the move $v_k \xrightarrow{d} C$, all disks smaller than d (except those residing on pegs v_1, \dots, v_{k-1} at that time) reside on peg C' , where $C' = A$ if $C = D$ and $C' = D$ otherwise. Replace all moves of disk d in the interval $[First(d), v_k \xrightarrow{d} C)$ with the single move $S \xrightarrow{d} v_k$ and perform it just before the move $v_k \xrightarrow{d} C$.

In each of the preceding two cases, the resulting sequence is also an optimal solution to $HAN(H_k, n)$. In this new solution, during the first move of each of the disks d and $d+1$, either all disks smaller than it (except those residing on pegs v_1, \dots, v_{k-1} at that time) reside on peg A or they all reside on peg D . Furthermore, the new optimal solution is identical to the solution from which it is constructed in the interval $[First(d+1), Last(1)]$, which suffices for the inductive step. \square

We call a solution to $HAN(H_k, n)$ *regular* if it satisfies the two following conditions.

- (1) No disk moves are performed between the only two moves that disk n performs: $S \xrightarrow{n} v_k$ and $v_k \xrightarrow{n} D$.
- (2) During the first move of a disk, either all smaller disks which are not on pegs v_1, \dots, v_{k-1} reside on peg A or they all reside on peg D .

By Lemmas 3.5 and 3.10, there exists an optimal solution to $HAN(H_k, n)$ which is regular, denoted by OPT_{reg} . Proving that $|OPT_{reg}| \geq |MoveH_k(n)|$ would complete the proof of Theorem 3.3.

LEMMA 3.11. *In OPT_{reg} , the configuration during $S \xrightarrow{n} v_k$ is $(k-1)$ -occupied.*

PROOF. Since OPT_{reg} is regular, during the move $S \xrightarrow{n} v_k$ of disk n , p disks, denoted from smallest to largest by l_1, \dots, l_p , reside on pegs v_1, \dots, v_{k-1} , and $n-1-p$ disks, denoted from smallest to largest by d_1, \dots, d_{n-1-p} , reside on peg A . Each disk in $\{d_1, \dots, d_{n-1-p}\}$ must make at least two moves: one to v_k and another from v_k to A . It follows that the minimum number of moves needed to transfer these disks from S to A (disregarding moves to pegs v_1, \dots, v_{k-1}) is at least $2(n-1-p)$. The minimum number of moves needed to transfer these $n-1-p$ disks from A to D is no less than $h_{n-1-p}^{(aw_0)} = 3^{n-1-p} - 1$. Disk n makes two moves. The number of moves that disks l_1, \dots, l_p make to pegs v_1, \dots, v_{k-1} is $p(k-1)$. By Lemma 3.9, the minimum number of moves needed to transfer disks l_1, \dots, l_p from pegs v_1, \dots, v_{k-1} to D , disregarding moves to pegs v_1, \dots, v_{k-1} ,

is at least $\frac{3^p+2p-1}{2}$. Altogether,

$$h_n^{(k)} \geq 2(n-1-p) + 3^{n-1-p} - 1 + 2 + p(k-1) + \frac{3^p + 2p - 1}{2}.$$

By Lemma 3.7,

$$h_n^{(k)} \leq \frac{3^{n-k+1} + 2n + 3^{k-1} + 2 \cdot (k-1)^2 - 2}{2}.$$

Assume for contradiction that $p < k - 1$. Next, we show that

$$\begin{aligned} g(n, k, p) &= (2 \cdot 3^{n-1-p} + 4(n-1-p) + 3^p + 2p + 1 + 2p(k-1)) \\ &\quad - (3^{n-k+1} + 2n + 3^{k-1} + 2 \cdot (k-1)^2 - 2) > 0, \end{aligned} \quad (1)$$

yielding a contradiction.

Notice that $p \leq k - 2$, and so, $3^{n-1-p} \geq 3^{n-k+1}$. Hence,

$$g(n, k, p) \geq 3^{n-1-p} + 2n - 1 - 2p + 3^p - 3^{k-1} + 2(k-1)(p-k+1).$$

Clearly $g(n, k, p)$ is monotone increasing with n for all k and p , and so, proving that $g(2k-2, k, p) > 0$ for all k and $p \leq k-2$ is sufficient for obtaining the desired contradiction (1).

Define $h(k, p) = g(2k-2, k, p)$. It is easy to verify that $h(k, p)$ is monotone decreasing with p for all $p \leq k-2$, and so proving that $h(k, k-2) > 0$ would complete the proof. Indeed, $h(k, k-2) = 3^{k-2} + 1 > 0$, and we are done. \square

By Lemma 3.11, in OPT_{reg} , all pegs v_1, \dots, v_{k-1} are occupied during the first move $S \xrightarrow{n} v_k$ of disk n . Let l_i be the disk residing on peg v_i at this moment, $1 \leq i \leq k-1$. By Observation 3.6, $l_1 > l_2 > \dots > l_{k-1}$. Denote the $n-k+1$ disks in $\{1, \dots, n\} \setminus \{l_1, \dots, l_{k-1}\}$, from smallest to largest, by d_1, \dots, d_{n-k+1} . For convenience, we define $x = l_1$. (Note that $x+1 = d_{x-k+2}$.)

In what follows up to and including Lemma 3.21, we establish a lower bound on the number of moves in OPT_{reg} , as stated in the following proposition. The proof of Theorem 3.3 would easily follow.

PROPOSITION 3.12.

$$\begin{aligned} |OPT_{reg}| \geq & 2(x-k+1) + \frac{3^{n-k+1} + 2(n-k+1) - 1}{2} \\ & - \frac{3^{x-k+1} + 2(x-k+1) - 1}{2} - 3^{x-k+1} + 1 + (k-1)^2 \\ & + \frac{3^{k-2} + 2(k-2) - 1}{2} + 2 + 3^{x-1} - 1. \end{aligned}$$

Observation 3.13. In OPT_{reg} , for each $1 \leq i < j \leq n$, $First(i)$ is performed before $First(j)$, and $Last(i)$ is performed after $Last(j)$.

Algorithm 6. OPT_{sub} :

-
- 1: $MoveH_1(|\{d_1, \dots, d_{x-k+1}\}|, S, \bar{C}, C)$
 - 2: OPT_{x+1}
 - 3: $MoveAux_0(|\{d_1, \dots, d_{x-k+1}\}|, A, v_1, D)$
-

LEMMA 3.14. *In OPT_{reg} , the first move of disk w is $S \xrightarrow{w} v_k$, for each $x+1 \leq w \leq n$. During this move, the disks l_1, \dots, l_{k-1} reside on the pegs v_1, \dots, v_{k-1} , respectively, and either all disks in $\{1, \dots, w-1\} \setminus \{l_1, \dots, l_{k-1}\}$ reside on peg A or they all reside on peg D .*

PROOF. Assume for contradiction that the first move of some disk w , $x+1 \leq w \leq n$, is to v_1 . By Lemma 3.11, during the first move $S \xrightarrow{n} v_k$ of disk n , a disk of size larger than x would reside on peg v_1 , a contradiction. It follows that all pegs v_1, \dots, v_{k-1} remain unchanged in the interval $[First(x+1), First(n))$. Hence, during $S \xrightarrow{w} v_k$, the disks l_1, \dots, l_{k-1} reside on the pegs v_1, \dots, v_{k-1} , respectively. The rest of the lemma follows from Lemma 3.10. \square

We use Claim 3.15, Lemma 3.16, and Corollary 3.19 to establish a lower bound on the number of moves in $OPT_{reg}[First(1), Last(x)]$.

CLAIM 3.15. $|OPT_{reg}[First(1), First(x+1)]|_{\{d_1, \dots, d_{x-k+1}\}} \geq 2(x-k+1)$.

PROOF. By Lemma 3.14, during the first move $S \xrightarrow{x+1} v_k$ of disk $x+1 = d_{x-k+2}$, all disks d_1, \dots, d_{x-k+1} are gathered on peg C , where C is either A or D . Hence, in the interval $[First(1), First(x+1)]$, each disk in $\{d_1, \dots, d_{x-k+1}\}$ makes at least two moves: to v_k and from v_k to C . The required result follows. \square

LEMMA 3.16.

$$|OPT_{reg}[First(x+1), Last(x+1)]|_{\{d_1, \dots, d_{x-k+1}\}} \geq h_{n-k+1}^{(1)} - h_{x-k+1}^{(1)} - h_{x-k+1}^{(aux_0)}$$

PROOF. Let OPT_{x+1} be the move sequence obtained from $OPT_{reg}[First(x+1), Last(x+1)]|_{\{d_1, \dots, d_{x-k+1}\}}$ by replacing v_k with v_1 in all disk moves. Define $m_1 = h_{n-k+1}^{(1)} - h_{x-k+1}^{(1)} - h_{x-k+1}^{(aux_0)}$. Next, we prove that $|OPT_{x+1}| \geq m_1$.

Observation 3.17. —By Lemma 3.14, in OPT_{x+1} , during the first move $S \xrightarrow{x+1} v_1$, either all disks d_1, \dots, d_{x-k+1} reside on peg A or they all reside on peg D . Denote this peg by C and the other peg by \bar{C} ; note that $\{C, \bar{C}\} = \{A, D\}$.

—In OPT_{x+1} , during the last move $v_1 \xrightarrow{x+1} D$, all disks d_1, \dots, d_{x-k+1} are gathered on peg A .

Assume for contradiction that the number of moves in OPT_{x+1} is some number m'_1 smaller than m_1 . Using OPT_{x+1} , we devise an algorithm that solves $HAN(H_1, |\{d_1, \dots, d_{x-k+1}\}|)$, denoted by OPT_{sub} .

Notice that in lines 1 and 3 of Algorithm OPT_{sub} , instead of just specifying the number $x-k+1$ of disks being moved, we write $|\{d_1, \dots, d_{x-k+1}\}|$ to (implicitly) indicate the corresponding disk set.

By Lemma 3.14 and Observation 3.17, it follows that OPT_{sub} indeed solves $HAN(H_1, |\{d_1, \dots, d_{n-k+1}\}|)$. It also holds that

$$|OPT_{sub}| = h_{x-k+1}^{(1)} + m'_1 + h_{x-k+1}^{(aux_0)} < h_{x-k+1}^{(1)} + m_1 + h_{x-k+1}^{(aux_0)} = h_{n-k+1}^{(1)}.$$

However, $h_{n-k+1}^{(1)}$ is defined to be the minimum number of moves needed to solve $HAN(H_1, |\{d_1, \dots, d_{n-k+1}\}|)$, a contradiction. \square

Remark. Observe that $|MoveH_k(n)[First(x+1), Last(x+1)]| = h_{n-k+1}^{(1)} - h_{x-k+1}^{(1)} - h_{x-k+1}^{(aux_0)}$. By Lemma 3.16, it follows that

$$|OPT_{reg}[First(x+1), Last(x+1)]| \geq |MoveH_k(n)[First(x+1), Last(x+1)]|.$$

CLAIM 3.18. (1) *The number of moves in $OPT_{reg}[First(1), Last(x)]$ that disks $x = l_1, \dots, l_{k-1}$ make to pegs v_1, \dots, v_{k-1} is $(k-1)^2$.*

(2) *The number of moves that are made in $OPT_{reg}[First(n), Last(x)]$ by disks $x = l_1, \dots, l_{k-1}$, disregarding moves to pegs v_1, \dots, v_{k-1} , is at least $h_{k-2}^{(1)} + 2$.*

PROOF. (1) In $OPT_{reg}[First(1), Last(x)]$, for each $1 \leq i \leq k-1$, disk l_i must move to each peg in $\{v_1, \dots, v_{k-1}\}$, yielding a total of $(k-1)^2$ moves.

(2) Clearly, x must make the two following moves: $v_{k-1} \xrightarrow{x} v_k, v_k \xrightarrow{x} D$. During the first move $S \xrightarrow{n} v_k$ of disk n , a $(k-1)$ -occupied configuration is obtained, with disk l_i residing on peg v_i , for each $1 \leq i \leq k-1$. During the last move $v_k \xrightarrow{x} D$ of disk x , all disks $1, \dots, x-1$ are gathered on peg A . In particular, all disks l_2, \dots, l_{k-1} are gathered on peg A at this moment. By Lemma 3.9, at least $h_{k-2}^{(1)}$ moves of disks l_2, \dots, l_{k-1} are made in $OPT_{reg}[First(n), Last(x)]$, disregarding moves to pegs v_1, \dots, v_{k-1} . The required result follows. \square

COROLLARY 3.19. $|OPT_{reg}[First(1), Last(x)]|_{\{l_1, \dots, l_{k-1}\}} \geq (k-1)^2 + h_{k-2}^{(1)} + 2$.

The following claim establishes a lower bound on $|OPT_{reg}(Last(x), Last(1))|$. Its proof is left to the reader.

CLAIM 3.20. $|OPT_{reg}(Last(x), Last(1))|_{\{1, \dots, x-1\}} \geq h_{x-1}^{(aux_0)}$.

Proposition 3.12 follows immediately from the following lemma.

LEMMA 3.21.

$$|OPT_{reg}| \geq 2(x-k+1) + h_{n-k+1}^{(1)} - h_{x-k+1}^{(1)} - h_{x-k+1}^{(aux_0)} + (k-1)^2 + h_{k-2}^{(1)} + 2 + h_{x-1}^{(aux_0)}.$$

PROOF. We have

$$|OPT_{reg}| = |OPT_{reg}[First(1), Last(x)]| + |OPT_{reg}(Last(x), Last(1))|.$$

Observe that

$$\begin{aligned} |OPT_{reg}[First(1), Last(x)]| &\geq |OPT_{reg}[First(1), First(x+1)]|_{\{d_1, \dots, d_{x-k+1}\}} \\ &\quad + |OPT_{reg}[First(x+1), Last(x+1)]|_{\{d_1, \dots, d_{n-k+1}\}} \\ &\quad + |OPT_{reg}[First(1), Last(x)]|_{\{l_1, \dots, l_{k-1}\}}. \end{aligned}$$

By Claim 3.15, Lemma 3.16, and Corollary 3.19, we have

$$|OPT_{reg}[First(1), Last(x)]| \geq 2(x - k + 1) + h_{n-k+1}^{(1)} - h_{x-k+1}^{(1)} - h_{x-k+1}^{(aux_0)} + (k - 1)^2 + h_{k-2}^{(1)} + 2.$$

By Claim 3.20,

$$|OPT_{reg}(Last(x), Last(1))| \geq h_{x-1}^{(aux_0)}.$$

Altogether,

$$|OPT_{reg}| \geq 2(x - k + 1) + h_{n-k+1}^{(1)} - h_{x-k+1}^{(1)} - h_{x-k+1}^{(aux_0)} + (k - 1)^2 + h_{k-2}^{(1)} + 2 + h_{x-1}^{(aux_0)}. \quad \square$$

Now we are ready to prove Theorem 3.3.

By Proposition 3.12,

$$\begin{aligned} h_n^{(k)} \geq f_1(n, x, k) &= 2(x - k + 1) + \frac{3^{n-k+1} + 2(n - k + 1) - 1}{2} \\ &\quad - \frac{3^{x-k+1} + 2(x - k + 1) - 1}{2} - 3^{x-k+1} + 1 + (k - 1)^2 \\ &\quad + \frac{3^{k-2} + 2(k - 2) - 1}{2} + 2 + 3^{x-1} - 1. \end{aligned}$$

By Lemma 3.7,

$$h_n^{(k)} \leq f_2(n, x, k) = \frac{3^{n-k+1} + 2n + 3^{k-1} + 2 \cdot (k - 1)^2 - 2}{2}.$$

By Lemma 3.11, $x \geq k - 1$. We prove the theorem by showing that for $x \geq k - 1$, $f_1(n, x, k) - f_2(n, x, k) \geq 0$. Define

$$\begin{aligned} g(x, k) &= f_1(n, x, k) - f_2(n, x, k) = 2(x - k + 1) + 3^{x-1} \\ &\quad - \frac{3^{x-k+2} + 3^{k-1} - 3^{k-2} + 2(x - k + 1) - 3}{2}. \end{aligned}$$

Notice that $g(x, k)$ is independent of n . It is easy to verify that $g(x, k)$ is monotone increasing with x , and so showing that $g(k - 1, k) \geq 0$ would imply the required result. Indeed,

$$g(k - 1, k) = 3^{k-2} - \frac{3^1 + 2 \cdot 3^{k-2} - 3}{2} = 0.$$

Theorem 3.3 follows: $h_n^{(k)} = \frac{3^{n-k+1} + 2n + 3^{k-1} + 2(k - 1)^2 - 2}{2}$.

4. The Infinite Family Γ'_C

In this section we define $H_k = C_k$, $Aux_0 = H_0$. All other definitions are as in Section 3.

4.1. MAIN RESULTS. All algorithms of Section 3 remain intact, except for Algorithms $MoveH_0(n)$ and $MoveAux_0(n)$, which are now changed to be the well-known optimal algorithm for the classical 3-peg problem, requiring 2^{n-1} moves. Denote the Hanoi graph depicted in Figure 3(b) by H_0 .

The following two theorems are the analogs of Theorems 3.1 and 3.2 of Section 3, and are proven similarly.

THEOREM 4.1. *An optimal solution to $HAN(H_1, n)$ is provided by $MoveH_1(n)$, and it takes $2^n + n - 1$ moves.*

THEOREM 4.2. *An optimal solution to $HAN(H_2, n)$ is provided by $MoveH_2(n)$, and it takes $\frac{2^n + 6n - 1 - (n \bmod 2)}{3}$ moves.*

The proof of the following theorem is more delicate.

THEOREM 4.3. *An optimal solution to $HAN(H_k, n)$ is provided by $MoveH_k(n)$ for any $k \geq 1$ and $n \geq 2k - 2$. For $k \leq 2$, it is optimal for all values of n . The number of moves it takes is*

$$\begin{cases} \frac{2^n + 6n - 1 - (n \bmod 2)}{3}, & k = 2, \\ 2^{n-k+1} + n + 2^{k-1} + (k-1)^2 - 2, & k = 1, k \geq 3. \end{cases}$$

The following observation implies that the move sequences $MoveH_1(n)$, $MoveH_2(n)$, and $MoveH_k(n)$ of this section are different from those of Section 3. This is clearly reflected in the difference in length of the respective move sequences.

Observation 4.4. —Algorithm $MoveAux_0(n)$ of this section is inherently different from that of Section 3.

—Algorithm $MoveAux_0(n)$ is embedded in Algorithms $MoveH_1(n)$, $MoveH_2(n)$, and $MoveH_k(n)$.

4.2. PROOF OF THEOREM 4.3. The following lemma is analogous, both in statement and in proof, to Lemma 3.7.

LEMMA 4.5. *Let $n \geq 2k - 2, k \geq 3$. Then Algorithm $MoveH_k(n)$ solves $HAN(H_k, n)$ in $2^{n-k+1} + n + 2^{k-1} + (k-1)^2 - 2$ moves.*

Fact 3.4, Lemma 3.5, Lemma 3.9, Lemma 3.10, Lemma 3.14, Lemma 3.16, Lemma 3.21, Claim 3.15, Claim 3.18, Claim 3.20, and Corollary 3.19 of Section 3 remain valid here, and are proven similarly.

Lemma 3.11 of Section 3 holds here as well.

LEMMA 4.6. *In OPT_{reg} , the configuration during $S \xrightarrow{n} v_k$ is $(k-1)$ -occupied.*

Our proof of Lemma 4.6 for $n \geq 2k - 2$ is very complicated and thus omitted. We hereby provide a simpler proof that holds only for sufficiently large n .

We use the following statement to prove the lemma.

CLAIM 4.7. $h_n^{(k)} \geq 2^{n-k} + 2n - 1$.

PROOF. Consider an optimal solution to $HAN(H_k, n)$ which is regular. By Lemma 3.10, the first move of disk n is $S \xrightarrow{n} v_k$. During this move, $t \leq k - 1$ disks, denoted from smallest to largest by l'_1, \dots, l'_t , reside on pegs v_1, \dots, v_{k-1} , and $n - 1 - t$ disks, denoted from smallest to largest by d'_1, \dots, d'_{n-1-t} , reside

on peg A . Each disk in $\{d'_1, \dots, d'_{n-1-t}\}$ must make at least two moves: one to v_k and another from v_k to A . It follows that the minimum number of moves needed to transfer these disks from S to A (disregarding moves to pegs v_1, \dots, v_{k-1}) is at least $2(n-1-t)$. The minimum number of moves needed to transfer these $n-1-t$ disks from A to D is no less than $h_{n-1-t}^{(aux_0)} = 2^{n-1-t} - 1$. Each disk in $\{l'_1, \dots, l'_t\}$ must make at least two moves: one to v_k and another from v_k to D . It follows that the minimum number of moves needed to transfer these disks from S to D (disregarding moves to pegs v_1, \dots, v_{k-1}) is at least $2t$. Disk n makes two moves. Altogether, at least $2^{n-1-t} - 1 + 2(n-1-t) + 2t + 2 \geq 2^{n-k} + 2n - 1$ moves are made. \square

PROOF OF LEMMA 4.6. During the first move $S \xrightarrow{n} v_k$ of disk n , p disks, denoted from smallest to largest by l_1, \dots, l_p , reside on pegs v_1, \dots, v_{k-1} , and $n-1-p$ disks, denoted from smallest to largest by d_1, \dots, d_{n-1-p} , reside on peg A . By Claim 4.7, $2^{n-1-p-k} + 2(n-1-p) - 1$ moves are needed to transfer disks d_1, \dots, d_{n-1-p} from S to A . The minimum number of moves needed to transfer these $n-1-p$ disks from A to D is no less than $h_{n-1-p}^{(aux_0)} = 2^{n-1-p} - 1$. Disk n makes two moves. The number of moves that disks l_1, \dots, l_p make to pegs v_1, \dots, v_{k-1} is $p(k-1)$. By Lemma 3.9, the minimum number of moves needed to transfer disks l_1, \dots, l_p from pegs v_1, \dots, v_{k-1} to D (disregarding moves to pegs v_1, \dots, v_{k-1}) is at least $h_p^{(1)} = 2^p + p - 1$. Altogether,

$$\begin{aligned} h_n^{(k)} &\geq 2^{n-1-p-k} + 2(n-1-p) - 1 + 2^{n-1-p} - 1 + 2 + p(k-1) + 2^p + p - 1 \\ &= 2^{n-1-p} + 2^{n-1-p-k} + 2^p + 2n - p - 3 + p(k-1). \end{aligned}$$

By Lemma 4.5,

$$h_n^{(k)} \leq 2^{n-k+1} + 2n + 2^{k-1} + (k-1)^2 - 2.$$

Assume for contradiction that $p < k-1$. Next, we show that

$$\begin{aligned} g(n, k, p) &= (2^{n-1-p} + 2^{n-1-p-k} + 2^p + 2n - p - 3 + p(k-1)) \\ &\quad - (2^{n-k+1} + 2n + 2^{k-1} + (k-1)^2 - 2) > 0, \end{aligned}$$

yielding a contradiction. Notice that $p \leq k-2$, and so, $2^{n-1-p} \geq 2^{n-k+1}$ and $2^{n-1-p-k} \geq 2^{n-2k+1}$. Hence,

$$\begin{aligned} g(n, k, p) &\geq 2^{n-k+1} + 2^{n-2k+1} + 2^0 - 2^{n-k+1} - 2^{k-1} - 1 - (k-1)^2 - k + 2 \\ &= 2^{n-2k+1} - 2^{k-1} - k(k-1) + 1. \end{aligned}$$

Clearly, for sufficiently large n , $g(n, k, p) > 0$. \square

Proposition 3.12 is restated here as follows.

PROPOSITION 4.8.

$$|OPT_{reg}| \geq 2^{n-k+1} + 2^{x-1} - 2^{x-k+2} + 2^{k-2} + (k-1)^2 + n + x - k + 1.$$

Now we are ready to prove Theorem 4.3. This proof is analogous to the proof of Theorem 3.3.

By Proposition 4.8,

$$\begin{aligned} h_n^{(k)} &\geq f_1(n, x, k) \\ &= 2^{n-k+1} + 2^{x-1} - 2^{x-k+2} + 2^{k-2} + (k-1)^2 + n + x - k + 1. \end{aligned}$$

By Lemma 4.5,

$$h_n^{(k)} \leq f_2(n, x, k) = 2^{n-k+1} + n + 2^{k-1} + (k-1)^2 - 2.$$

By Lemma 4.6, $x \geq k-1$. We prove the theorem by showing that for $x \geq k-1$, $f_1(n, x, k) - f_2(n, x, k) \geq 0$. Define

$$\begin{aligned} g(x, k) &= f_1(n, x, k) - f_2(n, x, k) \\ &= 2^{x-1} - 2^{x-k+2} + 2^{k-2} - 2^{k-1} + x - k + 3. \end{aligned}$$

Notice that $g(x, k)$ is independent of n . It is easy to verify that $g(x, k)$ is monotone increasing with x , and so showing that $g(k-1, k) \geq 0$ would imply the required result. Indeed,

$$g(k-1, k) = 2^{k-2} - 2^1 + 2^{k-2} - 2^{k-1} + 2 = 0.$$

Theorem 4.3 follows: $h_n^{(k)} = 2^{n-k+1} + n + 2^{k-1} + (k-1)^2 - 2$.

5. A Generalization to the Wider Family— Γ'

Sapir [2004] devised a unified algorithm for all strongly connected Hanoi graphs on three vertices, and proved its optimality. In our notation, Sapir provided an optimal algorithm for $\Gamma'|_0$, denoted by $UnifiedMoveH_0(n) = UnifiedMoveH_0(n, S, A, D)$.

For the case $k = 1$, continuing the work of Sapir [2004], we provide an optimal unified algorithm for $\Gamma'|_1$, denoted by $UnifiedMoveH_1(n) = UnifiedMoveH_1(n, S, A, D)$. The move sequence $UnifiedMoveH_1(n)$ is obtained from $UnifiedMoveH_0(n)$ by replacing the first move $S \xrightarrow{i} C$ of disk i , where C is either A or D , with the two following moves: $S \xrightarrow{i} v_1$, $v_1 \xrightarrow{i} C$, for all $1 \leq i \leq n$. The optimality of $UnifiedMoveH_1(n)$ is implied by the optimality of $UnifiedMoveH_0(n)$ and the following lemma, whose proof is left to the reader.

LEMMA 5.1. *Let H_1 be in $\Gamma'|_1$. For all n , there exists an optimal solution to $HAN(H_1, n)$, such that for each $1 \leq i \leq n$, no other disk moves are performed between the first two moves of disk i .*

For a Hanoi graph G , denote by $Dist_G(S, D)$ the distance from S to D in G . The following lemma, whose proof is omitted, holds for all graphs in $\{\Gamma'|_k : k \geq 1\}$. We use it to prove Theorem 5.3.

LEMMA 5.2. *Let H_k be in $\Gamma'|_k$, $k \geq 1$. For all n , in any optimal solution to $HAN(H_k, n)$ the largest disk performs $Dist_{H_k}(S, D)$ moves. Furthermore, there exists an optimal solution in which no other disk moves are performed between the first two moves of the largest disk.*

Let $UnifiedMoveH_1^k(n)$ be the move sequence obtained from $UnifiedMoveH_1(n)$ by replacing v_1 with v_k in all disk moves. It is easy to see that $UnifiedMoveH_1^k(n)$ is a feasible solution to $\Gamma'|_k$, for all $k \geq 1$.

Algorithm 7. *UnifiedMoveH₂(n, S, A, D)*:

The algorithm moves n disks of distinct sizes on any graph in $\Gamma'|_2$. The disks are initially stacked on S and are moved to D , using v_1, v_2 , and A as auxiliary pegs.

```

1: if  $n = 1$  then
2:    $S \xrightarrow{1} v_2$ 
3:   UnifiedMoveH0(1,  $v_2$ ,  $A$ ,  $D$ )
4: else if  $n \geq 2$  then
5:   if there is an arc from  $v_2$  to  $D$  then
6:     UnifiedMoveH2( $n - 2$ ,  $S$ ,  $D$ ,  $A$ )
7:      $S \xrightarrow{n-1} v_1$ 
8:      $S \xrightarrow{n} v_2$ ;  $v_2 \xrightarrow{n} D$ 
9:      $v_1 \xrightarrow{n-1} v_2$ ;  $v_2 \xrightarrow{n-1} D$ 
10:    UnifiedMoveH0( $n - 2$ ,  $A$ ,  $v_2$ ,  $D$ )
11:   else {there is an arc from  $A$  to  $D$ }
12:      $S \xrightarrow{1} v_1$ 
13:     UnifiedMoveH1( $\{|2, \dots, n - 1|\}$ ,  $S$ ,  $A$ ,  $D$ )
14:      $S \xrightarrow{n} v_2$ ;  $v_2 \xrightarrow{n} A$ 
15:     UnifiedMoveH0( $\{|2, \dots, n - 1|\}$ ,  $D$ ,  $A$ ,  $v_2$ )
16:      $A \xrightarrow{n} D$ 
17:     UnifiedMoveH0( $\{|2, \dots, n - 1|\}$ ,  $v_2$ ,  $A$ ,  $D$ )
18:      $v_1 \xrightarrow{1} v_2$ ;  $v_2 \xrightarrow{1} A$ ;  $A \xrightarrow{1} D$ 
19:   end if
20: end if

```

For the case $k = 2$, we provide an optimal unified algorithm for $\Gamma'|_2$, denoted by $UnifiedMoveH_2(n) = UnifiedMoveH_2(n, S, A, D)$. Notice that in lines 13, 15, and 17 of Algorithm $UnifiedMoveH_2(n)$, instead of just specifying the number $n - 2$ of disks being moved, we write $\{|2, \dots, n - 1|\}$ to (implicitly) indicate the corresponding disk set.

The proof of the following theorem is substantially more involved than the corresponding proofs for $k = 0, 1$, and is omitted.

THEOREM 5.3. *For all n , $UnifiedMoveH_2(n)$ is an optimal algorithm for $\Gamma'|_2$.*

For the case $k \geq 3, n \geq 2k - 2$, we provide a unified algorithm for $\Gamma'|_k$, denoted by $UnifiedMoveH_k(n) = UnifiedMoveH_k(n, S, A, D)$, and conjecture its optimality.

Conjecture 5.4. *For all $k \geq 3$ and $n \geq 2k - 2$, $UnifiedMoveH_k(n)$ is an optimal algorithm for $\Gamma'|_k$.*

Let $UnifiedMoveP_k(k - 1, v_k, A, D)$ be the move sequence obtained from the move sequence $UnifiedMoveH_0(k - 1, v_k, A, D)$ by replacing the *first* move $v_k \xrightarrow{i} C$ of disk i , where C is either A or D , with the move sequence that carries disk i along the path (v_{k-i}, \dots, v_k, C) , for all $1 \leq i \leq k - 1$.

Observe that Algorithm $UnifiedMoveH_k(n)$ coincides with Algorithm $MoveH_k(n)$ of Section 3 on the two families $\{R_k : k \geq 3\}$ and $\{C_k : k \geq 3\}$.

Algorithm 8. *UnifiedMoveH_k(n, S, A, D):*

The algorithm moves n disks of distinct sizes on any graph in $\{\Gamma'_k : k \geq 3\}$. The disks are initially stacked on S and are moved to D , using v_1, \dots, v_k and A as auxiliary pegs.

```

1: for  $j = 1$  to  $k - 1$  do
2:   for  $l = 0$  to  $k - j - 1$  do
3:      $v_l \xrightarrow{j} v_{l+1}$ 
4:   end for
5: end for
6: UnifiedMoveH1k(n - k + 1, S, A, D)
7: UnifiedMovePk(k - 1, vk, A, D)

```

The definition of a regular solution is generalized. We say that a solution to $HAN(G, n)$, $G \in \Gamma'$, is *regular* if it satisfies the three following conditions.

- (1) Disk n performs $Dist_G(S, D)$ moves.
- (2) No disk moves are performed between the first two moves of disk n .
- (3) During the first move of a disk, either all smaller disks which are not on pegs v_1, \dots, v_{k-1} reside on peg A or they all reside on peg D .

Similarly to Sections 3 and 4, we prove that there exists an optimal solution which is regular, denoted by OPT_{reg} . We conjecture that the following two lemmas hold.

LEMMA 5.5. *In OPT_{reg} , the configuration during $S \xrightarrow{n} v_k$ is $(k - 1)$ -occupied.*

LEMMA 5.6. *In OPT_{reg} , during the move $S \xrightarrow{n} v_k$, peg v_i contains disk $k - i$, for each $1 \leq i \leq k - 1$.*

We believe that a process similar to that shown in Sections 3 and 4 would produce the proofs of these two lemmas and imply the validity of our conjecture.

6. Future Research

The most evident open problem is to either prove or give a counterexample to Conjecture 5.4. We conclude the article by outlining additional directions for future research.

(1) In the Tower of Hanoi literature, it is commonly assumed that the number k of pegs is fixed, whereas the number n of disks may be arbitrarily large. Hence our assumption of $n \geq 2k - 2$, for $k \geq 3$, is reasonable. Nonetheless, it would be combinatorially interesting to study the complementary case of $n \leq 2k - 3$, $k \geq 3$. The proof of Theorem 3.3 holds true until Lemma 3.11, which ceases to hold when $n \leq 2k - 3$ (the same holds with respect to Theorem 4.3 and Lemma 4.6).

The set $\{v_1, \dots, v_{k-1}\}$ of pegs can be viewed as a “temporary storage device,” with each peg in it holding at most one disk. Intuitively, during the first move of disk n , the storage device is to be occupied in full, assuming that n is sufficiently large. Lemma 3.11 follows this intuition, asserting that for $n \geq 2k - 2$, all pegs v_1, \dots, v_{k-1} are occupied during the first move of disk n . This statement does not hold, though, for smaller values of n . Occupying all pegs v_1, \dots, v_{k-1}

simultaneously is impossible for $n \leq k - 2$, whereas for $k - 1 \leq n \leq 2k - 3$, even though possible, can be (extremely) wasteful (the reader is encouraged to verify this in the particular case $n = k = 3$). To see where the proof of Lemma 3.11 fails in the case $n \leq 2k - 3$, note that it no longer holds that $g(n, k, p) > 0$.

Using the storage device is tempting: A disk stored in it does not interfere with disk moves outside of it. However, the storage device is not “free of charge”: A disk stored in it has to pay for this luxury with the $k - 1$ moves that carry this disk along the path $(S = v_0, \dots, v_{k-1})$. Determining which portion of the storage device is to be occupied during the first move of disk n , as a function of k and n , for all $k \geq 3$ and $n \leq 2k - 3$, may be a challenging problem by itself. Having resolved this vital issue, the rest of the proof should follow similar lines as those in the proof of Theorem 3.3.

(2) In a sense, the reduced Hanoi graphs comprising Γ' are the *sparsest* possible. Consider a general member G_k in $\Gamma'|_k$. Removing any arc of the strongly connected component yields either a different member in $\Gamma'|_k$ or a nonsolvable Hanoi graph (see Theorem A in Section 1). Removing any arc (v_i, v_{i+1}) of the path P_k yields a nonreduced graph, as the vertices v_1, \dots, v_{k-1} become inessential. Finally, the graph G'_k obtained by removing the arc (S, v_k) reduces to the degenerate graph $G = G_0$. To see this, note that in order to get from S to D without taking the shortcut (S, v_k) , each disk is prone to move through the entire path P_k . Hence, given an optimal solution to G , an optimal solution to G'_k is obtained from it by replacing the first move $S \xrightarrow{i} C$ of disk i , where C is either A or D , with the move sequence that carries i along the path $(S = v_0, \dots, v_k, C)$, for all $1 \leq i \leq n$.

Being a family of sparsest possible graphs (in the sense explained before), Γ' serves as a fundamental candidate for study. This follows the intuition that the denser the graph, the more difficult it is to find an optimal solution to it, as there is more freedom at each move. A challenging problem along these lines is to study an infinite family of *dense* graphs in Γ .

(3) Another appealing direction for future research is to expand Γ' to include analogous graphs on the vertex set $\{A, B, S = v_0, \dots, v_k = D\}$, with B replacing the former destination peg D and v_k serving as the new destination peg. We anticipate that our techniques would be also applicable to this wider family of graphs.

ACKNOWLEDGMENTS. The authors thank D. Berend for setting the preliminary variant of the problem studied in this article. The authors are grateful to D. Berend, Y. Dinitz, and A. Sapir for substantial help in improving the presentation of the work.

REFERENCES

- ALLOUCHE, J. P., ASTOORIAN, D., RANDALL, J., AND SHALLIT, J. O. 1994. Morphisms, squarefree strings, and the Tower of Hanoi puzzle. *Amer. Math. Monthly* 101, 651–658.
- AZRIEL, D., AND BEREND, D. 2006. On a question of Leiss regarding the Hanoi Tower problem. *Theor. Comput. Sci.* 369, 1-3, 377–383.
- CHEN, X., AND SHEN, J. 2004. On the Frame-Stewart conjecture about the Towers Of Hanoi. *Siam J. Comput.* 33, 584–589.
- DINITZ, Y., AND SOLOMON, S. 2006. Optimal algorithms for Tower of Hanoi problems with relaxed placement rules. In *Proceedings of the 17th International Symposium on Algorithms and Computation (ISAAC)*, T. Asano, Ed. Lecture Notes in Computer Science, vol. 4288, Springer, 36–47.
- DINITZ, Y., AND SOLOMON, S. 2008. Optimality of an algorithm solving the Bottleneck Tower of Hanoi problem. *ACM Trans. Algor.* 4, 3, Art. # 15.

- DUDENEY, H. E. 1907. *The Canterbury Puzzles (and Other Curious Problems)*. Thomas Nelson and Sons, London.
- ER, M. C. 1985. The complexity of the generalised cyclic Towers of Hanoi. *J. Algor.* 6, 351–358.
- FRAME, J. S. 1941. Solution to advanced problem 3918. *Amer. Math. Monthly* 48, 216–217.
- HINZ, A. M., 1992. Pascal’s triangle and the Tower of Hanoi. *Amer. Math. Monthly* 99, 538–544.
- KORF, R. E., AND FELNER, A. 2007. Recent progress in heuristic search: A case study of the four-peg Towers of Hanoi problem. In *Proceedings of the 20th International Joint Conferences on Artificial Intelligence (IJCAI)*, M. M. Veloso, Ed. International Joint Conferences on Artificial Intelligence, Menlo Park, CA, 2324–2329.
- LEISS, E. L. 1983. Solving the ‘Towers of Hanoi’ on graphs. *J. Combin. Inf. Syst. Sci.* 8, 81–89.
- LEISS, E. L. 1984. Finite Hanoi problems: How many discs can be handled? *Congr. Numer.* 44, 221–229.
- LUCAS, E. 1893. *Récréations Mathématiques*. Vol. III. Gauthier-Villars, Paris.
- LUNNON, W. F. 1986. The Reve’s puzzle. *The Comput. J.* 29, 478.
- MINSKER, S. 1989. The Towers of Hanoi rainbow problem: Coloring the rings. *J. Algor.* 10, 1–19.
- POOLE, D. 1992. The Bottleneck Towers of Hanoi problem. *J. Recreational Math.* 24, 3, 203–207.
- SAPIR, A. 2004. The Towers of Hanoi with forbidden moves. *The Comput. J.* 47, 20–24.
- SCORER, R. S., GRUNDY, P. M., AND SMITH, C. A. B. 1944. Some binary games. *The Math. Gazette* 280, 96–103.
- STEWART, B. M. 1939. Advanced problem 3918. *Amer. Math. Monthly* 46, 363.
- STEWART, B. M. 1941. Solution to advanced problem 3918. *Amer. Math. Monthly* 48, 217–219.
- STOCKMEYER, P. K. 1994. Variations on the four-post Tower of Hanoi puzzle. *Congr. Numer.* 102, 3–12.
- SZEGEDY, M. 1999. In how many steps the k peg version of the Towers of Hanoi game can be solved? In *Proceedings of the 16th Symposium on Theoretical Aspects of Computer Science (STACS)*, C. Meinel and S. Tison, Eds. Lecture Notes in Computer Science, vol. 1563, Springer, 356–361.
- WOOD, D. 1981. The Towers of Brahma and Hanoi revisited. *J. Recreational Math.* 14, 1, 17–24.

RECEIVED MARCH 2006; REVISED JUNE 2008; ACCEPTED JULY 2008