SHORTEST PATHS IN THE TOWER OF HANOI GRAPH AND FINITE AUTOMATA

DAN ROMIK

Abstract. We present efficient algorithms for constructing a shortest path between two configurations in the Tower of Hanoi graph, and for computing the length of the shortest path. The key element is a finite-state machine which decides, after examining on the average only a small number of the largest discs (asymptotically, $\frac{63}{38} \approx 1.66$), whether the largest disc will be moved once or twice. This solves a problem raised by Andreas Hinz, and results in a better understanding of how the shortest path is determined. Our algorithm for computing the length of the shortest path is typically about twice as fast as the existing algorithm. We also use our results to give a new derivation of the average distance $\frac{466}{885}$ between two random points on the Sierpiński gasket of unit side.

Key words. Tower of Hanoi, finite automata, Sierpiński gasket.

AMS subject classifications. 68R05, 28A80.

1. Introduction.

The *Tower of Hanoi* puzzle, invented in 1883 by the French mathematician Edouard Lucas, has become a classic example in the analysis of algorithms and discrete mathematical structures (see e.g. [3, section 1.1]). The puzzle consists of n discs, no two of the same size, stacked on three vertical pegs, in such a way that no disc lies on top of a smaller disc. A permissible *move* is to take the top disc from one of the pegs and move it to one of the other pegs, as long as it is not placed on top of a smaller disc. The set of configurations of the puzzle, together with the permissible moves, thus forms a graph in a natural way. The number of vertices in the n-disc Hanoi graph is 3^n.

The main question of interest is to find *shortest paths* in the configuration graph, i.e., shortest sequences of moves leading from a given initial configuration to a given terminal configuration. The simplest and most well-known case is that in which it is required to move all the discs from one of the pegs to another, i.e. where the initial and terminal configurations are two of the three "perfect" configurations with all the discs on the same peg. This is very easy, and can be shown to take exactly $2^n - 1$ moves. More difficult is to get from a given arbitrary initial configuration to one of the perfect configurations - Hinz [6] calls this the "p1" problem. This takes $2^n - 1$ moves in the worst case (which is, for example, when the initial configuration is another perfect configuration), and on the average $\frac{2}{3} \cdot (2^n - 1)$ moves for a randomly chosen initial configuration [4]. Moreover, there is a simple and efficient algorithm to compute the shortest path in this case.

In the most general case of arbitrary initial *and* terminal configurations, however, the question of computing the shortest path and its length (the "p2" problem [6]) in the most efficient manner, has not been completely resolved so far. (The worst-case behavior is still $2^n - 1$ moves, and the average number of moves for random initial and terminal configurations has been shown [2],[5] to be asymptotically $(1 + o(1))\frac{466}{885} \cdot 2^n$.) The main obstacle in the understanding of the behavior of the shortest path, has been the behavior of the largest disc that "separates" the initial and terminal configurations, i.e. the largest disc which is not on the same peg in both configurations (trivially, any larger discs may simply be ignored). It is not difficult to see [6] that in a shortest path, this disc will be moved either once (from the source peg to the target peg) or twice (from the source to the target, via the third peg). The problem is to decide which of the two alternatives is the correct one. Once this is settled, the path may be constructed by two applications of the algorithm for the p1 problem. Hinz [6] proposed an algorithm for the computation of the shortest path based on this idea. The algorithm consists essentially of computing the length of the path for both alternatives and choosing the shorter of the two.

In this paper, we propose a more thorough explanation of the process whereby it is decided which of the two paths is the shortest. We show that it is possible to keep track of the relevant information using a finite-state machine, which at each step reads the locations of the next-smaller disc in the initial and terminal configurations, and changes its internal state accordingly. Eventually, the machine reaches a terminal state, whereupon it pronounces which of the two paths is the shortest. For a random input, its expected stopping time is computed to be $\frac{63}{38}$, asymptotically when the number of discs grows to infinity. In other words, after observing on the average the locations of just the ≈ 1.66 largest discs in the initial and terminal configurations, we will know which of the paths to choose, and we will be able to continue using the algorithm for the p1 problem. If one is interested just in the length of the shortest path, then our algorithm is typically about twice as fast as the algorithm proposed by Hinz [6] (with a small constant overhead due to the initial 1.66 discs), since it overrides the need to compute both the distance for the path that moves the largest disc once, and the path that moves it twice.

The paper is organized as follows: In the next section, we define the *discrete Sierpiński gasket* graph, a graph which is isomorphic to the Tower of Hanoi configuration graph, but for which the labeling of the vertices is simpler to understand. In section 3, we present the main ideas for the discrete Sierpiński gasket graph, and then in section 4 show how to translate the results to the Hanoi graph by a re-labeling of the vertices. In section 5 we perform a probabilistic analysis of the finite-state machine, to compute the average number $\frac{63}{38}$ of discs that need to be read in order to decide whether the largest disc will be moved once or twice, and to give a new derivation of the asymptotic value $(1+o(1))\frac{466}{885}\cdot 2^n$ for the average distance between two random configurations in the n-disc Hanoi graph, or equivalently of the statement that the average shortest-path distance between two random points in the Sierpiński gasket of unit side is equal to $\frac{466}{885}$. In section 6 we discuss extensions and some open problems.

2. The discrete Sierpiński gasket. We now define a family of graphs called *discrete Sierpiński gaskets*. These graphs are finite versions of the famous fractal constructed by the Polish mathematician Waclaw Sierpiński in 1915. The connection between the Tower of Hanoi problem and the Sierpiński gasket was first observed by Ian Stewart [12], and was later used by Andreas Hinz and Andreas Schief [9] in their calculation of the average distance between points on the Sierpiński gasket. The discrete Sierpiński gasket graphs that we define are identical to the graphs $S(n,3)$ defined by Klavzar and Milutinovic in [10], and similar (although this requires proof) to the graphs S_n defined in [9], so some of the discussion below parallels the discussion in those papers.

The nth discrete Sierpiński gasket graph, which we denote by SG_n, consists of the vertex set $V(SG_n) = \{T,L,R\}^n$ (the symbols T,L,R indicate "top", "left" and "right", respectively), with the edges defined as follows: First, for each $x = a_{n-1}a_{n-2}...a_1a_0 \in V(SG_n)$ (for reasons that will become apparent below, this will be our standard indexing of the coordinates of the vertices of SG_n) we have edges connecting x to

$$a_{n-1}a_{n-2}...a_1\beta, \qquad \beta \in \{T,L,R\} \setminus \{a_0\}$$

Second, define the *tail* of $x = a_{n-1}a_{n-2}...a_0$ as the suffix $a_k a_{k-1}...a_1 a_0$ of x, where k is maximal such that $a_k = a_{k-1} = ... = a_0$. If x has a tail of length $k+1 < n$, then x is of the form $a_{n-1}a_{n-2}...a_{k+2}\beta\alpha\alpha...\alpha$, in which case, connect x with an edge to the vertex

$$a_{n-1}a_{n-2}...a_{k+2}\alpha\beta\beta...\beta$$

One possible embedding of SG_n in the plane is illustrated in Figure 2.1 below. This embedding makes clear the meaning of the labeling of the vertices: The first letter (the "most significant digit") signifies whether the vertex is in the top, left or right triangles inside the big triangle; the next letter locates the vertex within the top, left or right thirds of that triangle, etc.

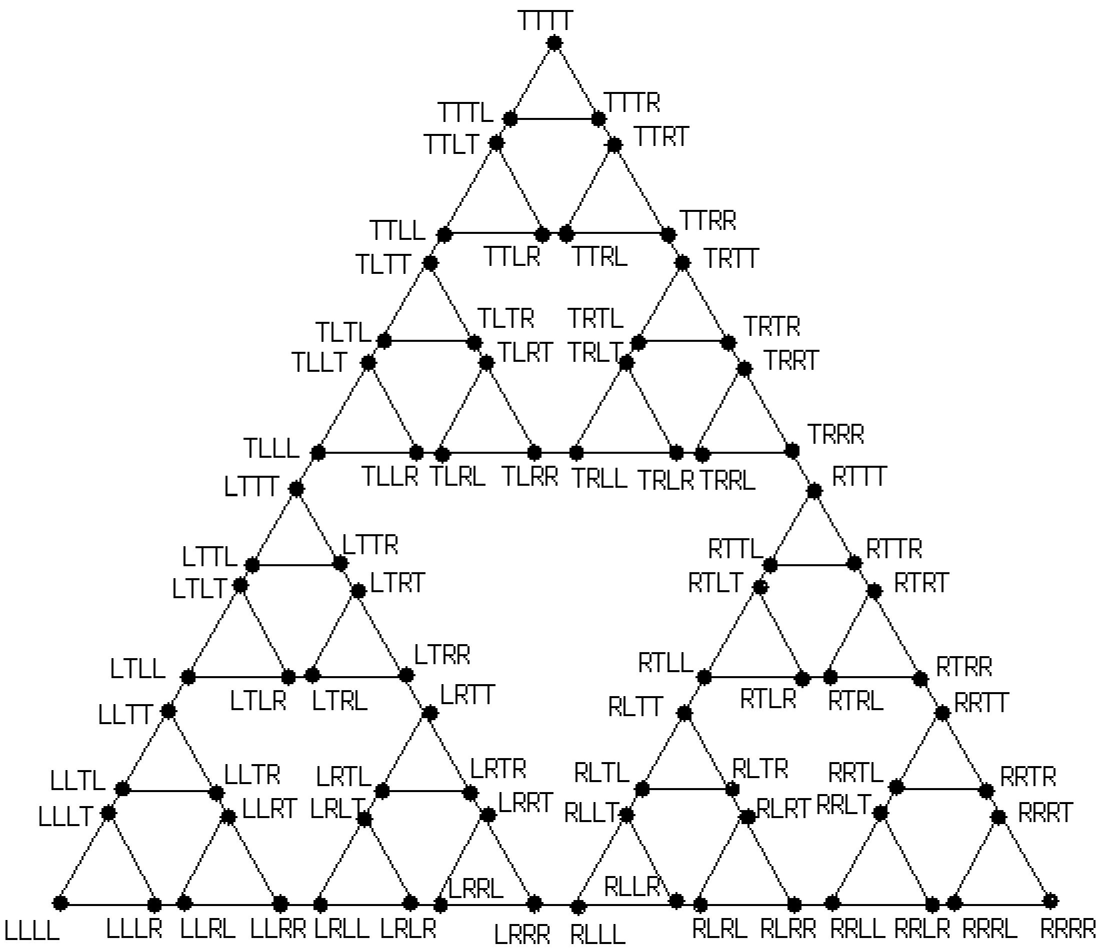

FIG. 2.1. *The graph SG_4*

It will be shown in section 4 that SG_n is isomorphic, in a computationally straightforward way, to the n-disc Hanoi graph. (The same was shown in [10], with less emphasis on explicit computation of the isomorphism.) Thus, the problem of shortest paths on the Hanoi graph reduces to that of shortest paths in the discrete Sierpiński gasket. We tackle this problem in the next section.

3. Shortest paths in SG_n. For vertices $x, y \in V(SG_n)$, we define the distance $d(x, y)$ to be the length of a shortest path from x to y. Our goal is to write down a recursion equation for this distance, which is at the heart of the finite-state machine we will construct to compute $d(x, y)$. First, let us review briefly some of the known facts about $d(x, y)$ in the simple case when y is one of the "perfect" configurations $LLL...L, RR...R, TT...T$. For concreteness, assume that $y = LLL...L$, and let $x = a_{n-1}a_{n-2}...a_1a_0 \in V(SG_n)$ as before. Then it is known that

$$d(x, y) = \sum_{a_k \neq L} 2^k$$

A simple algorithm exists for computing a shortest path from x to y in this case. In the Hanoi labeling of the graph, the algorithm is described in [6]. In the current labeling, the algorithm is even simpler and is based on the binary number system: if one identifies the symbol L with 0 and the symbols R and T with 1, then traversing the edges of the graph becomes equivalent to the operations of subtraction or addition of 1 in binary notation. The number of steps to reach $LL...L \equiv 00...0$ is then clearly the right-hand side in the above equation.

With these preparatory remarks, we now attack the problem of general $x = a_{n-1}a_{n-2}...a_0, \quad y = b_{n-1}b_{n-2}...b_0$. First, observe that we may assume that $a_{n-1} \neq b_{n-1}$, since otherwise we may simply consider x and y as vertices in the graph SG_{n-1} (note the self-similar structure in the definition of the graph, also apparent in the Tower of Hanoi puzzle when one ignores the largest disc). For concreteness, we begin by analyzing in detail the case where $a_{n-1} = T, \quad b_{n-1} = R$. Referring to Figure 2.1 for convenience, we see that

$$d(x, y) = \min\Big(1 + d(x, TRRR...R) + d(y, RTT...T),$$

$$1 + 2^{n-1} + d(x, TLLL...L) + d(y, RLL...L)\Big),$$

since in a shortest path from x to y, one must go from the top triangle to the right triangle either through the edge $\{TRR...R, RTT...T\}$ (we call this Alternative 1, see Theorem 3.1 below) or through a shortest path from $LTT...T$ to $LRR...R$ (Alternative 2) - in the Tower of Hanoi language, this is an indication of the fact that in a shortest sequence of moves the largest disc must move either once or twice, see [6].

To simplify the next few equations, introduce the following notation: if $u = c_{n-1}c_{n-2}...c_0 \in \{T, L, R\}^n$, let $u' = c_{n-2}c_{n-3}...c_0$, and define for any $\alpha \in \{L, T, R\}$

$$f_\alpha(u) = \sum_{c_k \neq \alpha} 2^k$$

Then we have

$$d(x, y) = 1 + \min\Big(f_R(x') + f_T(y'), \quad 2^{n-1} + f_L(x') + f_L(y')\Big)$$

The recursion equations which will enable us to construct our finite-state machine and compute $d(x, y)$ are now given by the following theorem:

THEOREM 3.1 (The finite state machine). *For* $u = c_{n-1}c_{n-2}...c_0,\ \ v = d_{n-1}d_{n-2}...d_0 \in \{T, L, R\}^n$*, define the functions*

$$p(u,v) = \min\left(f_R(u) + f_T(v),\ 2^n + f_L(u) + f_L(v)\right)$$

$$q(u,v) = \min\left(2^n + f_R(u) + f_T(v),\ f_L(u) + f_L(v)\right)$$

$$r(u,v) = \min\left(f_R(u) + f_T(v),\ f_L(u) + f_L(v)\right)$$

(note that p, q, r *depend implicitly on the length* n *of the strings.) Then we have the equations*

$$p(u,v) = \begin{cases} f_R(u) + f_T(v) & \begin{array}{l} c_{n-1} = R, d_{n-1} = T \text{ or} \\ c_{n-1} = R, d_{n-1} = L \text{ or} \\ c_{n-1} = R, d_{n-1} = R \text{ or} \\ c_{n-1} = L, d_{n-1} = T \text{ or} \\ c_{n-1} = T, d_{n-1} = T \text{ or} \\ c_{n-1} = T, d_{n-1} = R \end{array} \quad \text{(Alternative 1)} \\ 2^n + p(u',v') & \begin{array}{l} c_{n-1} = T, d_{n-1} = L \text{ or} \\ c_{n-1} = L, d_{n-1} = R \end{array} \\ 2^n + r(u',v') & c_{n-1} = L, d_{n-1} = L \end{cases}$$

$$q(u,v) = \begin{cases} f_L(u) + f_L(v) & \begin{array}{l} c_{n-1} = L, d_{n-1} = L \text{ or} \\ c_{n-1} = L, d_{n-1} = T \text{ or} \\ c_{n-1} = L, d_{n-1} = R \text{ or} \\ c_{n-1} = R, d_{n-1} = L \text{ or} \\ c_{n-1} = T, d_{n-1} = L \text{ or} \\ c_{n-1} = T, d_{n-1} = R \end{array} \quad \text{(Alternative 2)} \\ 2^n + q(u',v') & \begin{array}{l} c_{n-1} = T, d_{n-1} = T \text{ or} \\ c_{n-1} = R, d_{n-1} = R \end{array} \\ 2^n + r(u',v') & c_{n-1} = R, d_{n-1} = T \end{cases}$$

$$r(u,v) = \begin{cases} f_R(u)+f_T(v) & c_{n-1}=R, d_{n-1}=T \qquad \textit{(Alternative 1)} \\ f_L(u)+f_L(v) & c_{n-1}=L, d_{n-1}=L \qquad \textit{(Alternative 2)} \\ 2^{n-1}+r(u',v') & c_{n-1}=L, d_{n-1}=T \text{ or } c_{n-1}=R, d_{n-1}=L \\ 2^{n}+r(u',v') & c_{n-1}=T, d_{n-1}=R \\ 2^{n-1}+p(u',v') & c_{n-1}=R, d_{n-1}=R \text{ or } c_{n-1}=T, d_{n-1}=T \\ 2^{n-1}+q(u',v') & c_{n-1}=T, d_{n-1}=L \text{ or } c_{n-1}=L, d_{n-1}=R \end{cases}$$

Alternatives 1,2 in the parentheses signify whether the minimum is attained by its first or second arguments, respectively. These equations will hold even for $n=1$ if one sets trivially for $u,v=\emptyset\in\{T,L,R\}^0=\{\emptyset\}$

$$f_\alpha(u)=0,$$

$$p(u,v)=0 \textit{ (Alternative 1)}, \tag{3.1}$$

$$q(u,v)=0 \textit{ (Alternative 2)}, \tag{3.2}$$

$$r(u,v)=0 \textit{ (tie)}. \tag{3.3}$$

Proof. First, note that if $\alpha\in\{T,L,R\}$ and $w\in\{T,L,R\}^n$ then trivially $f_\alpha(w)\le 2^n-1$.

Here is the proof of the equation for $p(u,v)$ in several sample cases; the full proof is a slightly tedious case by case verification and consists of similar computations, so we omit it.

Sample case 1: Assume that $(c_{n-1},d_{n-1})=(R,T)$. In that case, we have

$$\begin{aligned} p(u,v) &= \min\Big(f_R(u')+f_T(v'), 2^n+2^{n-1}+2^{n-1}+f_L(u')+f_L(v')\Big) \\ &= \min\Big(f_R(u')+f_T(v'), 2^{n+1}+f_L(u')+f_L(v')\Big) \\ &= f_R(u')+f_T(v') = f_R(u)+f_T(v), \end{aligned}$$

since $f_R(u')+f_T(v')\le 2^{n-1}-1+2^{n-1}-1<2^{n+1}$ so the minimum can only be attained by the first argument.

Sample case 2: Assume that $(c_{n-1},d_{n-1})=(R,L)$. Then we have

$$\begin{aligned} p(u,v) &= \min\Big(2^{n-1}+f_R(u')+f_T(v'), 2^n+2^{n-1}+f_L(u')+f_L(v')\Big) \\ &= 2^{n-1}+\min\Big(f_R(u')+f_T(v'), 2^n+f_L(u')+f_L(v')\Big) \\ &= 2^{n-1}+f_R(u')+f_T(v') = f_R(u)+f_T(v), \end{aligned}$$

again since $f_R(u')+f_T(v')\le 2^n-2<2^n$, so again Alternative 1 must hold.

Sample case 3: Assume that $(c_{n-1}, d_{n-1}) = (T, L)$. Then

$$\begin{aligned} p(u,v) &= \min\left(2^{n-1} + 2^{n-1} + f_R(u') + f_T(v'), 2^n + 2^{n-1} + f_L(u') + f_L(v')\right) \\ &= 2^n + \min\left(f_R(u') + f_T(v'), 2^{n-1} + f_L(u') + f_L(v')\right) \\ &= 2^n + p(u', v'). \end{aligned}$$

Note that the order of the arguments in the minimum is preserved, so that once the correct alternative for $p(u', v')$ is determined, this is propagated back to $p(u, v)$.
Sample case 4: Assume that $(c_{n-1}, d_{n-1}) = (L, L)$. Then

$$\begin{aligned} p(u,v) &= \min\left(2^{n-1} + 2^{n-1} + f_R(u') + f_T(v'), 2^n + f_L(u') + f_L(v')\right) \\ &= 2^n + r(u', v'). \end{aligned}$$

□

A schematic representation of the finite state machine is shown in Figures 3.1 and 3.2 below. We present two variants of the machine: the machine in Figure 3.1 only decides between Alternative 1 and Alternative 2, in the case in which x begins with the symbol T and y begins with R. The machine in Figure 3.2, which has auxiliary counters for the distance and for the variable n (so strictly speaking it is not really a finite-state automaton), actually computes $d(x, y)$, and it is designed to treat the general case of any two configurations $x, y \in V(SG_n)$. This is done by including an initial component that discards the first few symbols which are identical for x and y, and another component that permutes the symbols T, L, R to fit the design of the basic machine in Figure 3.1.

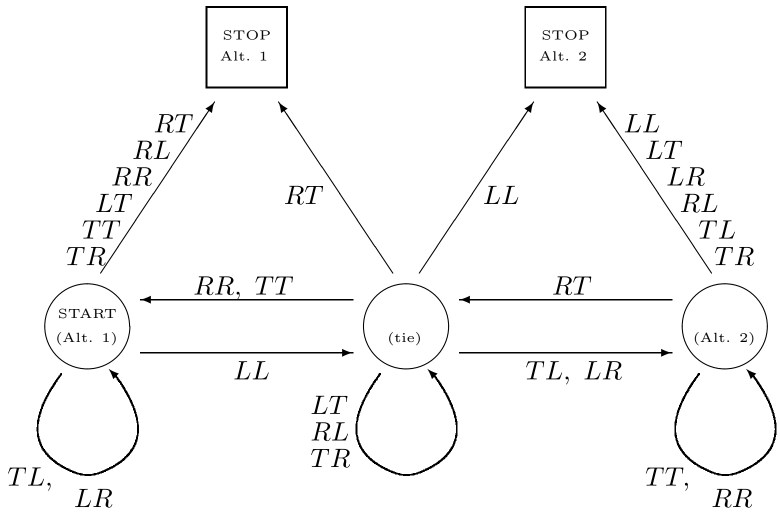

FIG. 3.1. *The finite state machine: deciding between Alternative 1 and Alternative 2. The two letters signify the two inputs from x and y, reading at each step the next-most-significant symbol. The parentheses in the non-terminal states indicate that if the input terminates without a decision, then in the* START *state Alternative 1 wins, in the rightmost state Alternative 2 wins, and in the middle state there is a tie, meaning that the shortest path is not unique and both alternatives are valid. (Termination of the input corresponds to the recursion equations leading to an evaluation of either of $p(u,v)$, $q(u,v)$ or $r(u,v)$ with $u = v = \emptyset$, so the above claim follows from equations* (3.1),(3.2),(3.3) *together with the fact mentioned in the proof of Theorem 3.1 that the order of the arguments is propagated throughout the recursion.)*

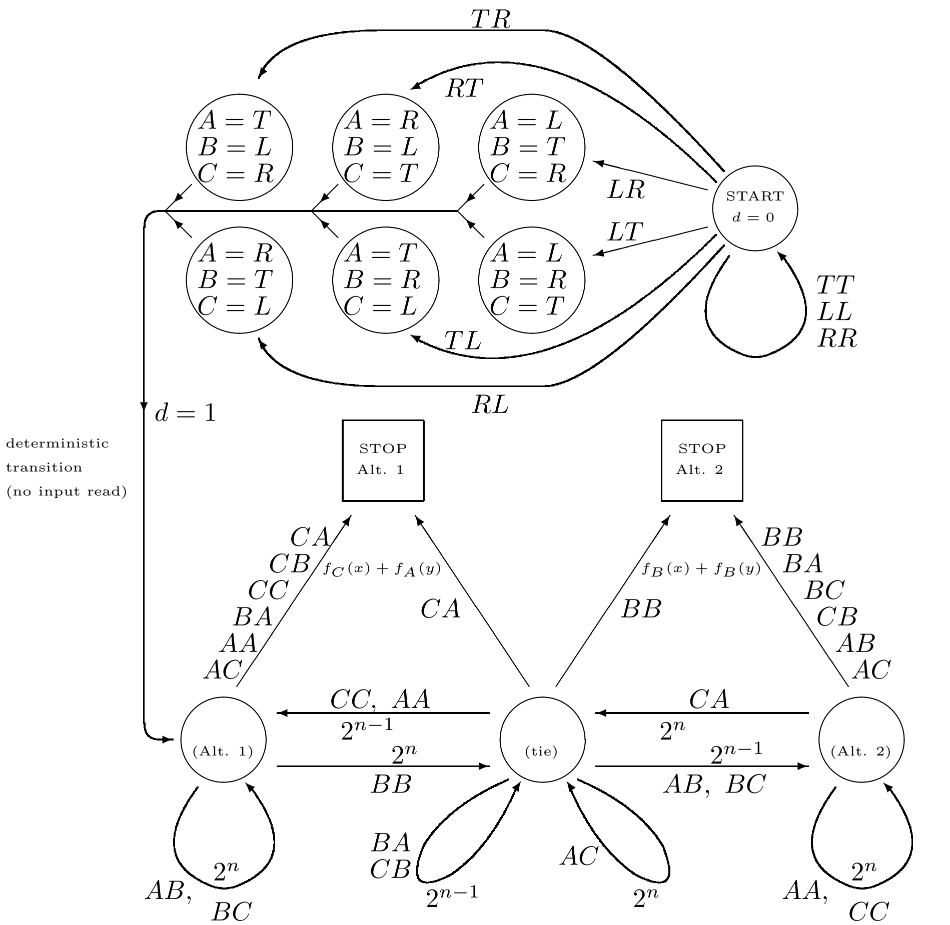

FIG. 3.2. *The finite state machine: computing $d(x, y)$, the general case. Add to d the number on each edge traversed, decrease n by 1 and replace x by x' and y by y'. For the deterministic transition, do not read input or decrease n.*

4. Translating between the Hanoi graph and SG_n. We now define the graph of configurations in the n-disc Tower of Hanoi puzzle, and show that it is isomorphic to SG_n. The isomorphism may be computed by reading sequentially the locations of the discs, starting with the largest one (which corresponds to the most significant digit in the Sierpiński gasket labeling), and following a diagram of permutations translating the labels of the three pegs into the symbols T, L, R (another finite-state machine!). Together with the results of the previous section, this will give an effective means of computing the length of the shortest path between any two vertices in the Hanoi graph, and of deciding whether the largest disc will be moved once or twice in a shortest path. After that, we describe briefly an algorithm for actually constructing the shortest path, based on the algorithm for getting to a perfect configuration.

Label the three pegs in the Tower of Hanoi with the symbols $0, 1, 2$. Since in a legal configuration, on each of the pegs the discs are arranged in increasing size from top to bottom, a configuration is described uniquely by specifying, for any disc, the label of its peg. Thus, we define H_n, the n-th *Hanoi graph*, to be the graph whose vertex set is the set $V(H_n) = \{0, 1, 2\}^n$ (with the coordinates of the vectors specifying, from left to right, the labels of the pegs of the largest disc, second-largest disc, etc.), and where edges between configurations correspond to permissible moves. Figure 4.1 shows the graph H_4.

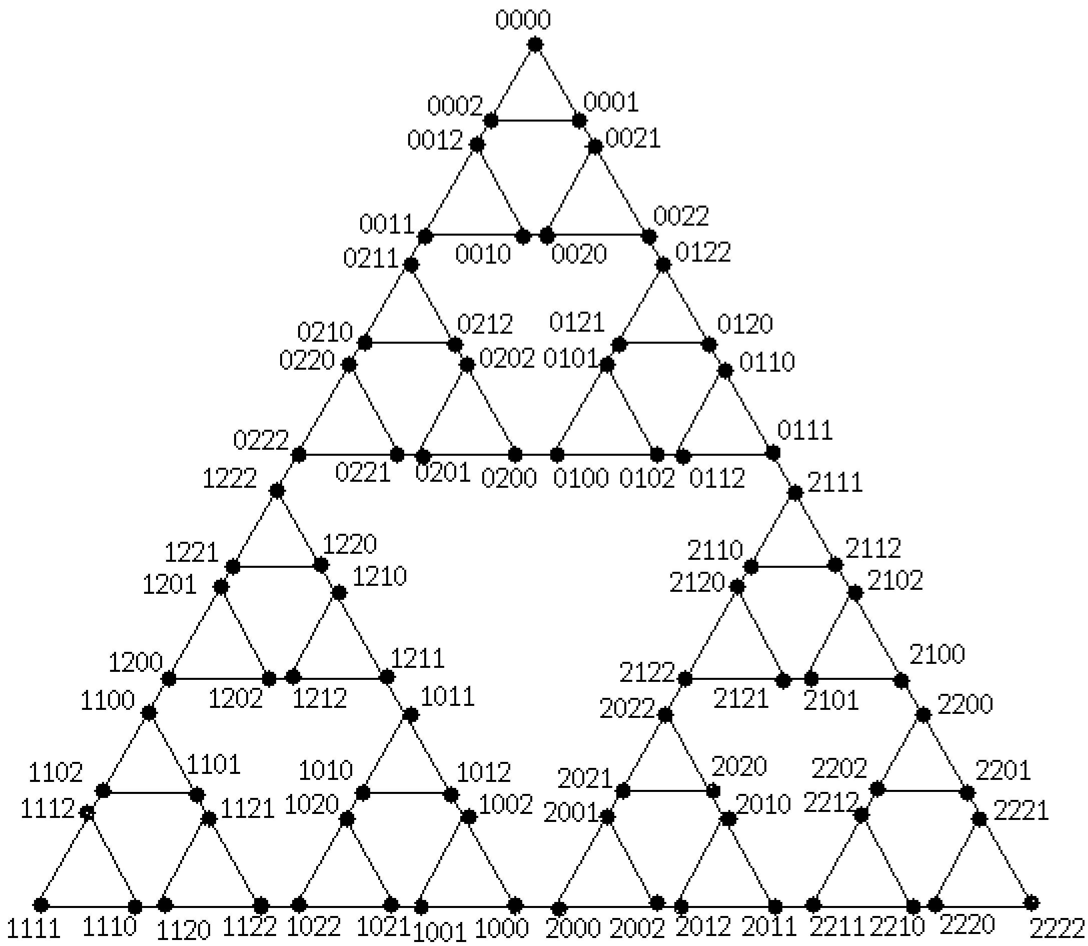

Fig. 4.1. *The graph* H_4

The isomorphism between H_n and SG_n is now described by the following theorem:

THEOREM 4.1. *H_n and SG_n are isomorphic graphs. The finite-state machine shown in Figure 4.2 below translates a Hanoi configuration $s \in \{0,1,2\}^n$ into a Sierpiński gasket labeling $z \in \{T,L,R\}^n$, by reading the digits from left to right and outputting the symbols T,L,R at each step according to the identifications in its internal state, then changing the internal state according to the input.*

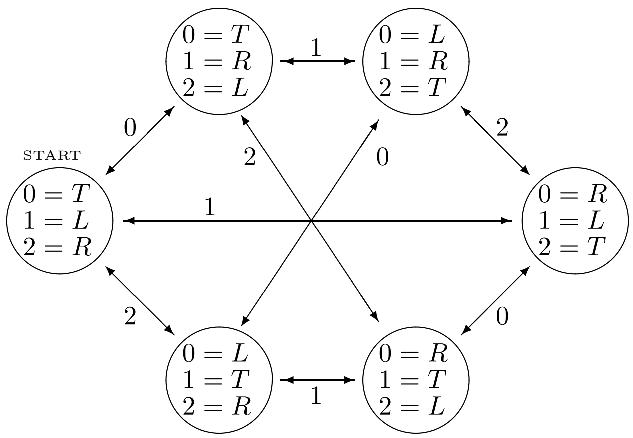

FIG. 4.2. *Computing the isomorphism between H_n and SG_n*

Proof. This is Theorem 2 in [10]. There it was claimed simply that H_n and SG_n are isomorphic, but the proof, which is by induction, actually describes how to compute the isomorphism, and this is easily seen to be equivalent to our finite-state machine formulation. (A similar argument is used in the proof of Lemma 2 in [9], which constructs an isomorphism between H_n and a different "discrete Sierpiński graph", defined in a geometrical way which is not obviously related to the current SG_n graph.) □

Summary.. By running the machines of Figures 3.2 and 4.2 in parallel, we now have an algorithm for computing $d(x,y)$ for two arbitrary configurations in the Hanoi graph, and for solving the decision problem for the largest disc, i.e. to decide whether the largest disc which it is necessary to move will move once or twice. As we will show in the next section, when x and y are randomly chosen configurations, the expected stopping time of the machine is 63/38. (This random variable even has an exponential tail distribution, so with very high probability only a small number of discs will need to be read to solve the decision problem.) Having solved the decision problem, the shortest path may now be computed in a straightforward manner, as described in [6], using the algorithm for getting to a perfect configuration (use the algorithm described in [6], or the algorithm for the Sierpiński gasket described in section 3 together with the machine of Figure 4.2 – which incidentally leads to an algorithm for getting to perfect configurations which we have not found in the literature).

5. The case of random inputs.

5.1. How many discs must be read to solve the decision problem?. In this section, we calculate the average number of discs that must be read in order to decide whether in a shortest path the largest disc will be moved once or twice. Let $x = a_{n-1}a_{n-2}...a_0 \in V(H_n)$, $y = b_{n-1}b_{n-2}...b_0 \in V(H_n)$. Assume that we have already discarded the largest discs which for x and y were on the same peg, so that $a_{n-1} \neq b_{n-1}$. The algorithm for solving the decision problem then tells us to run the machines of Figures 3.2 and 4.2 until they reach a terminal state (or we run out of input). Since we have already initialized by discarding irrelevant discs, we will really be using the machine of Figure 3.1 (keeping track of the correct identification of the symbols L, T, R with the pegs $0, 1, 2$). Since we are dealing with random inputs, what we are really interested in is the absorption time of the Markov chain whose transition matrix is

$$\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{pmatrix} 2/9 & 1/9 & 0 & 2/3 & 0 \\ 2/9 & 1/3 & 2/9 & 1/9 & 1/9 \\ 0 & 1/9 & 2/9 & 0 & 2/3 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

into the terminal states 4 and 5. We may identify these two states to get the simpler matrix

$$\begin{array}{c} 1 \\ 2 \\ 3 \\ (45) \end{array} \begin{pmatrix} 2/9 & 1/9 & 0 & 2/3 \\ 2/9 & 1/3 & 2/9 & 2/9 \\ 0 & 1/9 & 2/9 & 2/3 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

For $i = 1, 2, 3$, denote by t_i the expected time to get to state (45), *starting from state* i. Then clearly we have the equations

$$\begin{aligned} t_1 &= 1 + \frac{2}{9}t_1 + \frac{1}{9}t_2 \\ t_2 &= 1 + \frac{2}{9}t_1 + \frac{1}{3}t_2 + \frac{2}{9}t_3 \\ t_3 &= 1 \qquad\quad + \frac{1}{9}t_2 + \frac{2}{9}t_3 \end{aligned}$$

It may easily be verified that the solution to this system of equations is

$$t_1 = 63/38, \quad t_2 = 99/38, \quad t_3 = 63/38$$

The value $t_1 = 63/38$ is our expected stopping time, since $i = 1$ corresponds to the initial state. Note that this value is the limit as $n \to \infty$ of the average number of discs that must be read; in reality, for finite n the value will be slightly smaller since after n steps we run out of input and the machine terminates even if it has not reached a terminal state. To summarize:

THEOREM 5.1. *The decision problem for shortest paths can be solved in average time* $O(1)$. *Specifically, the average number of disc pairs that our algorithm must read, once identical discs have been discarded, is bounded from above by, and converges as* $n \to \infty$ *to,* 63/38.

5.2. The average distance between points on the Sierpiński gasket. Hinz and Schief [9] computed the average length 466/885 of a shortest path between two random points on the *infinite* Sierpiński gasket of unit side. An equivalent result of Hinz [5] and of Chan [2], in terms of the Tower of Hanoi, is that the average number of moves in a shortest path between two random configurations in the n-disc Tower of Hanoi, is asymptotically $(1 + o(1))(466/885) \cdot 2^n$ as $n \to \infty$.

Without going into too much detail, we show that it is possible to obtain the value of 466/885 just by looking at the finite-state machine of Figure 3.2. Since we are dealing with the infinite gasket, we start with $n = 0$ and, as before, decrease the value of n after each step, so that n will go into the negative integers. Let d_1, d_2, d_3, d_4 be the expected accumulated values of the variable d if one starts the machine, with initial values $n = 0, d = 0$, at either of the four non-terminal states, in order of their distance from the state START (so d_1 is the total distance; d_2 is the distance after discarding identical most-significant digits of x and y, etc.). Then we have the equations

$$\begin{aligned}
d_1 &= \frac{1}{3}\cdot\frac{1}{2}d_1 + \frac{2}{3}\cdot\frac{1}{2}d_2 \\
d_2 &= \frac{2}{9}\cdot\left(1+\frac{1}{2}d_2\right) + \frac{1}{9}\cdot\left(1+\frac{1}{2}d_3\right) + \frac{2}{3}\cdot\left(\frac{1}{2}+\frac{2}{3}\right) \\
d_3 &= \frac{2}{9}\cdot\left(\frac{1}{2}+\frac{1}{2}d_2\right) + \frac{2}{9}\cdot\left(\frac{1}{2}+\frac{1}{2}d_3\right) + \frac{1}{9}\cdot\left(1+\frac{1}{2}d_3\right) \\
&\quad + \frac{2}{9}\cdot\left(\frac{1}{2}+\frac{1}{2}d_4\right) + \frac{2}{9}\cdot\frac{2}{3} \\
d_4 &= \frac{1}{9}\cdot\left(1+\frac{1}{2}d_3\right) + \frac{2}{9}\cdot\left(1+\frac{1}{2}d_4\right) + \frac{2}{3}\cdot\left(\frac{1}{2}+\frac{2}{3}\right)
\end{aligned}$$

The value (1/2+2/3) in the second and fourth equations is the expected value of $f_C(x) + f_A(y)$ (respectively $f_B(x) + f_B(y)$), given that the first pair of inputs in the lower part of Figure 3.2 is one of the six values AC, AA, BA, CC, CB, CA (respectively BB, BA, BC, CB, AB, AC).

Again, it may be verified that the solution to this system of equations is

$$d_1 = \frac{466}{885}, \quad d_2 = \frac{233}{177}, \quad d_3 = \frac{188}{177}, \quad d_4 = \frac{233}{177}$$

which gives our claimed value for d_1.

6. Extensions and open problems. We mention possible connections of our work to other questions related to the Tower of Hanoi and to the study of fractal structures similar to the Sierpiński gasket.

- **Higher dimensional Sierpiński gaskets and other fractals.** For each $n \geq 2$, there is a fractal known as the Sierpiński gasket in $\mathbb{R}^n$ analogous to the Sierpiński gasket in $\mathbb{R}^2$. Bandt and Kuschel [1] showed that the average distance between two points in the Sierpiński gasket in $\mathbb{R}^n$ is equal to

$$\frac{n}{(2n+1)(n+1)}\left(2n - \frac{n^2-1}{n^3+7n^2+7n+9}\right).$$

  In this case, the problem is again of determining which of several parts of the gasket a shortest path between two given points should pass through. It seems very likely that one can construct a finite-state machine to solve this problem, and that the result of Bandt and Kuschel can be reproved using this method. More generally, one can ask similar questions for the class of post-critically finite (p.c.f.) fractals (see [1] for the definition), and it would be interesting to characterize the family of such fractals for which one can solve the shortest path problem using a finite-state machine, and to give a general method for constructing such a machine given the symmetries of the fractal. As an example, we have computed the average distance between two points in the modified Sierpiński gasket (in $\mathbb{R}^2$) which has side lengths 2, 2 and 1. It is equal to

$$\frac{147644401107013}{168923515522320} \approx 0.955.$$

We omit the computation, which is somewhat tedious and uses basically the same ideas as the ones presented here.

- **Non-unique shortest paths in H_n.** In a recent paper [8], Hinz et al. prove the following formula for the number a_n of pairs (x, y) of vertices in the Tower of Hanoi graph H_n for which there are two shortest paths connecting x and y:

$$a_n = \frac{3}{4\sqrt{17}}\left[\left(\sqrt{17}+1\right)\left(\frac{5+\sqrt{17}}{2}\right)^n - 2\sqrt{17}\cdot 3^n + \left(\sqrt{17}-1\right)\left(\frac{5-\sqrt{17}}{2}\right)^n\right].$$

Their proof of this formula makes use of Stern's diatomic sequence. However, as the authors point out, this formula can also be proved using our finite automaton, since basically a_n counts the number of paths in the graph of states of the automaton in Figure 3.2 leading from the state START to the state (tie). By writing down the adjacency matrix of the graph of states and diagonalizing, one can obtain the formula above.

A related result, not mentioned in [8] but easily seen to follow from the same ideas, is the following: Let S be the Sierpiński gasket fractal in $\mathbb{R}^2$. Let A be the subset of $S \times S$ consisting of all pairs (x, y) of points in S for which there are two shortest paths connecting x and y in S. Then the Hausdorff dimension of A is $\log[(5+\sqrt{17})/2]/\log 2$.

REFERENCES

[1] C. Bandt, T. Kuschel, Self-similar sets VIII. Average interior distance in some fractals. Measure theory (Oberwolfach, 1990). *Rend. Circ. Mat. Palermo (2)* Suppl. No. 28 (1992), 307–317.
[2] T. Chan, A statistical analysis of the Towers of Hanoi problem. *Internat. J. Comput. Math.* 28, 543–623 (1988)
[3] R. L. Graham, D. E. Knuth, O. Patashnik, Concrete Mathematics, 2nd. Ed. Addison-Wesley, 1994.
[4] M. C. Er, An analysis of the generalized Towers of Hanoi problem. *BIT* 23, 429–435 (1983).
[5] A. Hinz, The Tower of Hanoi. *Enseign. Math.* 35, 289–321 (1989).
[6] A. Hinz, Shortest paths between regular states of the Tower of Hanoi. *Inform. Sci.* 63, 173–181 (1992).
[7] A. Hinz, The Tower of Hanoi. Algebras and Combinatorics (ICAC'97, Hong Kong), 277–289. Springer, Singapore, 1999.
[8] A. Hinz, S. Klavzar, U. Milutinovic, D. Parisse, C. Petr, Metric properties of the Tower of Hanoi graphs and Stern's diatomic sequence. *Europ. J. Combin.* 26, 693–708, (2005)
[9] A. Hinz, A. Schief, The average distance on the Sierpiński gasket. *Probab. Th. and Rel. Fields* 87, 129–138 (1990).
[10] S. Klavzar, U. Milutinovic, Graphs $S(n,k)$ and a variant of the Tower of Hanoi problem. *Czechoslovak Math. J.* 47 (122), 95–104 (1997).
[11] S. Klavzar, U. Milutinovic, C. Petr, On the Frame-Stewart algorithm for the multi-peg Tower of Hanoi problem. *Disc. Appl. Math.* 120, 141–157 (2002).

[12] I. Stewart, Le lion, le lama et la laitue. *Pour la Science* 142, 102-107 (1989).
[13] P. K. Stockmeyer, The Tower of Hanoi: A historical survey and bibliography, version 0.2. Preprint (2001). Available electronically at `http://www.cs.wm.edu/~pkstoc/h_papers.html`

Author's contact information:

Department of Statistics
367 Evans Hall
University of California
Berkeley, CA 94720-3860
USA

Phone: (510)642-6154
Fax: (510) 642-7892

Email: `romik@stat.berkeley.edu`