Discrete Applied Mathematics 177 (2014) 111–120

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

An efficient algorithm to determine all shortest paths in Sierpiński graphs☆

Andreas M. Hinz [a,b,*], Caroline Holz auf der Heide [a]

[a] Faculty of Mathematics, Computer Science and Statistics, LMU Munich, Germany
[b] Faculty of Natural Sciences and Mathematics, University of Maribor, Slovenia

ARTICLE INFO

Article history:
Received 10 February 2014
Received in revised form 27 May 2014
Accepted 28 May 2014
Available online 23 June 2014

Keywords:
Shortest path algorithm
Automata
Sierpiński graphs
Switching Tower of Hanoi

ABSTRACT

In the *Switching Tower of Hanoi* interpretation of *Sierpiński graphs* S_p^n, the *P2 decision problem* is to find out whether the largest moving disc has to be transferred once or twice in a shortest path between two given states/vertices. We construct an essentially optimal algorithm thus extending Romik's approach for $p = 3$ to the general case. The algorithm makes use of three automata and the underlying theory includes a simple argument for the fact that there are at most two shortest paths between any two vertices. The total number of pairs leading to non-unique solutions is determined and employing a Markov chain argument it is shown that the number of input pairs needed for the decision is bounded above by a number independent of n. Elementary algorithms for the length of the shortest path(s) and the best first move/edge are also presented.

0. Introduction

Among the open problems in [6, Chapter 9] is the challenge to "design an automaton analogous to Romik's automaton for a 'P2 task' in S_p^n, $p \geq 4$". In fact, in [13] D. Romik presented an algorithm which determines the distance between any two vertices of the Sierpiński graph S_3^n by deciding about the number of necessary moves of the largest disc in an optimal solution of the corresponding task in the Switching Tower of Hanoi interpretation of that graph. This interpretation, together with a more general definition of *Sierpiński graphs* S_p^n, $p \in \mathbb{N}$ and $n \in \mathbb{N}_0$, had been proposed in [8] by S. Klavžar and U. Milutinović. The common source of these questions was an open problem announced in [4, p. 179] concerning the *P2 decision problem* for the Tower of Hanoi, represented by the *Hanoi graph* H_3^n, which in turn is isomorphic to S_3^n; cf. the discussions in [6, Sections 2.4 and 4.1]. In [2], L.L. Cristea and B. Steinsky investigated the decision problem on another related class of graphs, the *Sierpiński triangle graphs* (cf. [9, p. 463]), on the *Sierpiński triangle* itself and on their 3-dimensional generalizations. Fundamental metric properties of Sierpiński graphs can be found in [12,7].

We will now solve the P2 decision problem for all S_p^n, thereby using, if not otherwise stated, definitions and notations from [6].[1] More specifically, we will construct an algorithm which, given vertices s and t of S_p^n (we also call (s, t) a *task*), returns the index of the largest disc (LD) moved on an optimal (i.e. minimal length) s, t-path and the answer to the decision question whether this disc moves once or twice or if both strategies lead to a shortest path. In the latter two cases, the

☆ ©A.M. Hinz, C. Holz auf der Heide 2014.

* Corresponding author at: Faculty of Mathematics, Computer Science and Statistics, LMU Munich, Germany.
E-mail addresses: andreas.hinz@um.si, hinz@math.lmu.de (A.M. Hinz), caroline@holzaufderheide.de (C. Holz auf der Heide).

[1] Most notably, for $q \in \mathbb{N}_0$ we write $[q]$ and $[q]_0$ for the q-segments $\{1, \ldots, q\}$ and $\{0, \ldots, q-1\}$ of $\mathbb{N}$ and $\mathbb{N}_0$, respectively. Moreover, $\binom{K}{2}$ stands for the set of all subsets of size 2 of a set K.

http://dx.doi.org/10.1016/j.dam.2014.05.049
0166-218X/

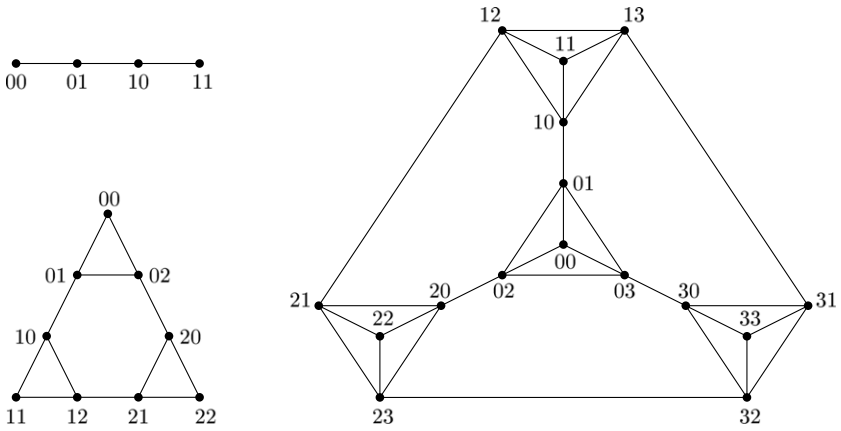

Fig. 1. The graphs S_2^2 (top left), S_3^2 (bottom left), and S_4^2 (right).

index of the subgraph different from initial and final subgraph (i.e. the *shortcut peg*) will also be detected. The goal of that algorithm is to obtain these output data with as little input as possible. With the information flowing from the algorithm it is then easy to determine the optimal edge from s, i.e. the *best first move* (BFM), and the distance $\mathrm{d}(s,t)$ of s and t in S_p^n, albeit using all information about these vertices. We begin in Section 1 with the theory, based on the methods in [8,13] and [6, Sections 2.4.3, 4.1 and 4.2.1], which will provide the correctness proof for the algorithm described in Section 2. In the concluding Section 3 we will analyse the complexity of the algorithm.

1. The theory

Just as the Tower of Hanoi game (cf. [6]), the *Switching Tower of Hanoi* is played with a certain number of moveable discs of different size stacked on some fixed vertical pegs. Any legal distribution of all discs among the pegs, i.e. with no larger disc lying on top of a smaller one, is called a (*regular*) *state*. Unlike in its famous archetype, where only one disc may be moved at a time, a move of the Switching Tower of Hanoi consists of the exchange of a topmost disc on one peg with the subtower of all smaller discs on top of another peg, including the case where the single disc is the smallest one and the corresponding subtower therefore empty. The P2 task is then, starting from a given arbitrary state, to find a (shortest) sequence of moves to get to another prescribed state.

As shown in [8, Theorem 1], the corresponding state graph, whose vertices are the states and the edges of which stand for the moves, is isomorphic to the *Sierpiński graph* S_p^n, where the *base* $p \in \mathbb{N}$ represents the number of pegs and the *exponent* $n \in \mathbb{N}_0$ is the number of discs. The graph is defined by

$$V(S_p^n) = [p]_0^n \quad \text{and} \quad E(S_p^n) = \left\{ \{\underline{s}ij^{d-1}, \underline{s}ji^{d-1}\} \mid \{i,j\} \in \binom{[p]_0}{2},\ d \in [n],\ \underline{s} \in [p]_0^{n-d} \right\};$$

here a vertex $s \in [p]_0^n$ is written in the form $s_n \dots s_1$ and describes a state where $s_d \in [p]_0$ is the label of the peg the disc $d \in [n]$ is lying on.[2] An edge can be viewed as the switch of disc d on peg i with the tower of $d-1$ smaller discs on peg j while the positions of larger discs, collected in $\underline{s}$, remain unchanged. For some examples of Sierpiński graphs, see Fig. 1.

An equivalent recursive definition [6, (4.7)] shows that the graphs S_p^n are connected; the canonical distance function will be denoted by d. The most important observation is[3]

$$\mathrm{d}(s, j^n) = \sum_{d=1}^{n} (s_d \neq j) \cdot 2^{d-1}, \tag{0}$$

where j^n is the *extreme vertex* corresponding to the *perfect state* when all discs lie on peg j. This is Proposition 4.5 of [6], where the proof can be found. Moreover, the shortest path from s to j^n is unique; cf. also [8, Lemma 4]. Uniqueness may be lost, however, for $p \geq 3$ and $n \geq 2$ if both initial state s and goal state t are arbitrary. (The case $p = 1$ or $n = 0$ is trivial because there is only one vertex and no edge, for $p = 2$ we have $S_2^n \cong P_{2^n}$, the path graph on 2^n vertices (cf. Fig. 1), and finally $S_p^1 \cong K_p$, the complete graph of order p, for any p.) A simplest example is with $s = 01$ and $t = 21$ in S_3^2, where two shortest paths 01, 02, 20, 21 and 01, 10, 12, 21 of length 3 differ by the number of moves of the LD, called LDMs, namely 1 or 2 (see Fig. 1). It is also possible that 2 LDMs are *necessary* on a shortest path as in the task (011, 211) in S_3^3. It will turn out that this is already the worst that can happen! (This is in strong contrast to the situation in general *Hanoi graphs* H_p^n, where up to

[2] Discs are numbered according to increasing size.

[3] We employ *Iverson's convention* that $(\mathfrak{S}) = 1$, if statement $\mathfrak{S}$ is true, and $(\mathfrak{S}) = 0$, if $\mathfrak{S}$ is false.

$p-1$ LDMs may be necessary; cf. [6, Theorem 5.43]. More on LDMs in Hanoi graphs can be found in [1].) The basic reasons are the "boxer rule" that the LD once removed from a peg will never return there [6, Lemma 4.7], the fact that the passage through two bypass subgraphs would take at least 2^n+1 moves and the value of the diameter which is $\mathrm{diam}(S_p^n)=2^n-1$. We summarize this in the following lemma (cf. [8, Theorem 5] or [6, Theorem 4.8]).

Lemma 1.1. *Let* $\mathbb{N}_0 \ni n < N \in \mathbb{N}$, $\underline{s} \in [p]_0^{N-n-1}$, $\{i,j\} \in \binom{[p]_0}{2}$, *and* $s, t \in [p]_0^n$. *Then in* S_p^N *we have*

$$\mathrm{d}(\underline{s}is, \underline{s}jt) = \min\{\mathrm{d}_k(is,jt) \mid k \in [p+1]_0 \setminus \{i,j\}\},$$

where $\mathrm{d}_k(is,jt) := \mathrm{d}(s,k^n)+1+2^n+\mathrm{d}(t,k^n)$ *for* $k \in [p]_0$ *and* $\mathrm{d}_p(is,jt) := \mathrm{d}(s,j^n)+1+\mathrm{d}(t,i^n)$; *the paths realizing these numbers of moves are unique and given by is* $\to ik^n \to ki^n \to kj^n \to jk^n \to jt$ *and is* $\to ij^n \to ji^n \to jt$, *respectively, where we have omitted the prefix* $\underline{s}$ *which remains constant throughout the paths. In particular, there are at most two LDMs in any optimal* s,t*-path in* S_p^N.

Remark 1.1. We have changed the exponent of the Sierpiński graph under investigation to N in order to emphasize the label, namely $n+1$, of the largest disc moved in a shortest path from $\underline{s}is$ to $\underline{s}jt$ in S_p^N and to make ready use of formula (0) in S_p^n.

We will now determine the value(s) of k for which the minimum in Lemma 1.1 is attained. For $p=1$ the lemma is void, and necessarily $k=p$ for $p=2$; so we may assume that $p \geq 3$ in the sequel. The method described in [6, Section 2.4.3] for $p=3$ can be extended to the general case in the following way. We first remark that by virtue of the boxer rule, the LD is disc $n+1$, such that we may assume that $\underline{s}$ is empty. Then, for $k \in [p]_0$,

$$\mathrm{d}_p(is,jt) \,\square\, \mathrm{d}_k(is,jt) \Leftrightarrow \rho := \mathrm{d}(s,j^n)-\mathrm{d}(s,k^n)+\mathrm{d}(t,i^n)-\mathrm{d}(t,k^n) \,\square\, 2^n,$$

where $\square \in \{<,=,>\}$. By (0), $\rho = \sum_{d=1}^n \beta_d \cdot 2^{d-1}$, with

$$\beta_d = (s_d=k)+(t_d=k)-(s_d=j)-(t_d=i) \in \{-2,-1,0,1,2\}.$$

Let us define $\rho_\nu = \sum_{d=1}^\nu \beta_d \cdot 2^{d-1}$ for $\nu \in [n+1]_0$; then $\rho_n=\rho$, $|\rho_\nu| < 2^{\nu+1}-1$, and $\rho_\nu - \rho_{\nu-1} = \beta_\nu \cdot 2^{\nu-1}$ for $\nu \neq 0$. This means that after the input of just one pair (i,j) of positions of the LD we cannot say anything yet, except when $n=0$, in which case there is only one disc which therefore has to be moved from i to j directly. So let us assume that $n \geq 1$ and put $g := s_n$ and $h := t_n$. Then

$$\beta_n = (g=k)+(h=k)-(g=j)-(h=i) \quad \text{and} \quad \rho \,\square\, 2^n \Leftrightarrow \rho_{n-1} \,\square\, (2-\beta_n)\cdot 2^{n-1}.$$

This leads to

Proposition 1.0. *If* $(g,h)=(i,j)$, $(j,\cdot)$ *or* $(\cdot,i)$, *then* $\square = <$ *for every* $k \in [p]_0 \setminus \{i,j\}$.
If $(g,h)=(k,k)$ *for some* $k \in [p]_0 \setminus \{i,j\}$, *then* $\rho \,\square\, 2^n \Leftrightarrow \rho_{n-1} \,\square\, 0$ *for* k.
If $(g,h)=(i,k)$ *or* (k,j) *for some* $k \in [p]_0 \setminus \{i,j\}$, *then* $\rho \,\square\, 2^n \Leftrightarrow \rho_{n-1} \,\square\, 2^{n-1}$ *for* k.
If $|\{g,h,i,j\}|=4$, *then* $\rho \,\square\, 2^n \Leftrightarrow \rho_{n-1} \,\square\, 2^{n-1}$ *for* $k=g$ *and* $k=h$.

This covers all possible inputs (g,h).

Proof. The different cases correspond to $\beta_n \leq 0$, $\beta_n = 2$, and $\beta_n = 1$ for the latter two, respectively. The fact that $\rho_{n-1} < 2^n-1$ covers the first case. In the second and third the value of k is uniquely determined. □

After the second input pair we can stop in the first case of Proposition 1.0 with the result that there is only one LDM and the shortest path is unique. We are also finished in the other cases if $n=1$ because $\rho_0=0$: for the second case we get a draw and for the last cases we again have only one LDM and a unique optimal solution. The reader can test the algorithm so far with the above example $s=01$ and $t=21$ in S_3^2, where $i=0, j=2$ and $g=1=h$, whence Proposition 1.0 tells us that $\square = =$ and that $k=1$ for the optimal solution with two LDMs. (The latter result is, of course, unavoidable since there are only three pegs present.)

Let us now assume that $n \geq 2$ and that Proposition 1.0 leaves us in the last case. Then we have

Proposition 1.1. *Let* $|\{g,h,i,j\}|=4$ *and* $\rho \,\square\, 2^n \Leftrightarrow \rho_\nu \,\square\, 2^\nu$ *for some* $\nu \in [n]_0$ *and either* $k=g$ *or* $k=h$. *Then for* $\ell \in [p]_0 \setminus \{g,h,j\}$ *and* $m \in [p]_0 \setminus \{g,h,i\}$,

if $\nu=0$ *or* $(s_\nu,t_\nu)=(j,\cdot)$, $(\cdot,i)$ *or* (ℓ,m) *we have* $\square = <$,
if $(s_\nu,t_\nu)=(k,k)$, *then* $\rho \,\square\, 2^n \Leftrightarrow \rho_{\nu-1} \,\square\, 0$ *for* k,
if $(s_\nu,t_\nu)=(\ell,k)$ *or* (k,m), *then* $\rho \,\square\, 2^n \Leftrightarrow \rho_{\nu-1} \,\square\, 2^{\nu-1}$ *for* k,
if $(s_\nu,t_\nu)=(g,h)$ *or* (h,g), *then* $\rho \,\square\, 2^n \Leftrightarrow \rho_{\nu-1} \,\square\, 2^{\nu-1}$ *for* $k=g$ *and* $k=h$.

This covers all possible inputs (s_ν,t_ν).

The *proof* is similar to the one for Proposition 1.0.

The possibly repeated application of Proposition 1.1 will eventually lead to the result $\square = <$ or it will fix the value of k. This shows in particular (cf. [8, Theorem 6]):

Corollary 1.1. *There are at most two optimal paths between any two vertices of a Sierpiński graph.*

Remark 1.2. If $\square = <$, the index for which $\min\{\mathrm{d}_k(is, jt) \mid k \in [p]_0 \setminus \{i, j\}\}$ is attained, might *not* be unique, as the simple examples of $is = 01, jt = 10$ in S_4^2 (see Fig. 1) or even $i = 0, j = 1$ in S_4^1 demonstrate.

We will now treat the cases for fixed $k \in \{g, h\} \setminus \{i, j\}$. We assume that $\ell, m \in [p]_0$ with $j \neq \ell \neq k \neq m \neq i$.

Proposition 1.2. A. *Let $\rho \square 2^n \Leftrightarrow \rho_\nu \square 2^\nu$ for some $\nu \in [n]_0$. Then*
if $\nu = 0$ or $(s_\nu, t_\nu) = (j, \cdot), (\cdot, i)$ or (ℓ, m), we have $\square = <$,
if $(s_\nu, t_\nu) = (k, k)$, then $\rho \square 2^n \Leftrightarrow \rho_{\nu-1} \square 0$,
if $(s_\nu, t_\nu) = (\ell, k)$ or (k, m), then $\rho \square 2^n \Leftrightarrow \rho_{\nu-1} \square 2^{\nu-1}$.
B. *Let $\rho \square 2^n \Leftrightarrow \rho_\nu \square 0$ for some $\nu \in [n]_0$. Then*
if $\nu = 0$, we have $\square = =$; otherwise
if $(s_\nu, t_\nu) = (j, i)$, we have $\square = <$,
if $(s_\nu, t_\nu) = (\ell, i)$ or (j, m), then $\rho \square 2^n \Leftrightarrow \rho_{\nu-1} \square 2^{\nu-1}$,
if $(s_\nu, t_\nu) = (k, k)$, we have $\square = >$,
if $(s_\nu, t_\nu) = (j, k), (k, i)$ or (ℓ, m), then $\rho \square 2^n \Leftrightarrow \rho_{\nu-1} \square 0$,
if $(s_\nu, t_\nu) = (\ell, k)$ or (k, m), then $\rho \square 2^n \Leftrightarrow \rho_{\nu-1} \square -2^{\nu-1}$.
C. *Let $\rho \square 2^n \Leftrightarrow \rho_\nu \square -2^\nu$ for some $\nu \in [n-1]_0$. Then*
if $\nu = 0$ or $(s_\nu, t_\nu) = (k, \cdot), (\cdot, k)$ or (ℓ, m), we have $\square = >$,
if $(s_\nu, t_\nu) = (j, i)$, then $\rho \square 2^n \Leftrightarrow \rho_{\nu-1} \square 0$,
if $(s_\nu, t_\nu) = (\ell, i)$ or (j, m), then $\rho \square 2^n \Leftrightarrow \rho_{\nu-1} \square -2^{\nu-1}$.

In every case all possible inputs (s_ν, t_ν) are covered.

Proof. The proof is again on the same lines as for Propositions 1.0 and 1.1. Note that in the last case of B a new type of comparison occurs which is then treated in C. □

Application of the preceding results yields the decision whether we have to use $b = 1$ or $b = 2$ LDMs for an optimal solution or if both options are possible (we write $b = 0$ then), and for $b \neq 1$ we are also given the shortcut peg k. This has been achieved using the minimum number possible of pairs of input (s_d, t_d) in each individual instance. We will analyse this performance in Section 3. Of course, we also get, albeit using all pairs, the distance $\mathrm{d}(\underline{s}is, \underline{s}jt) = \mathrm{d}(is, jt)$ using formula (0), namely

$$\begin{aligned} \mathrm{d}(is, jt) &= 2^{n+1} - 1 - \sum_{d=1}^{n} \{(s_d = j) + (t_d = i)\} \cdot 2^{d-1}, \qquad \text{if } b < 2, \\ \mathrm{d}(is, jt) &= 3 \cdot 2^n - 1 - \sum_{d=1}^{n} \{(s_d = k) + (t_d = k)\} \cdot 2^{d-1}, \quad \text{if } b = 2. \end{aligned} \tag{1}$$

Note that the sums σ in these formulas are in *hyperbinary representation* (cf. [6, p. 123]), i.e. of the form

$$\sigma_\beta = \sum_{d=1}^{n} \beta_d \cdot 2^{d-1}, \quad \beta = \beta_n \dots \beta_1 \in \{0, 1, 2\}^{[n]}.$$

They can be efficiently transformed into their ordinary binary representation

$$\sigma_\alpha = \sum_{d=0}^{n} \alpha_d \cdot 2^d, \quad \alpha = \alpha_n \dots \alpha_0 \in \{0, 1\}^{[n+1]_0} \setminus \{1^{n+1}\}$$

with the aid of $\gamma \in \{0, 1\}^{[n+1]_0}$, defined by

$$\gamma_0 = 0, \quad \forall\, d \in [n] : \ \gamma_d = (1 - \gamma_{d-1})(\beta_d = 2) + \gamma_{d-1}(\beta_d \neq 0), \tag{2}$$

whence

$$\alpha_n = \gamma_n, \quad \forall\, d \in [n] : \ \alpha_{d-1} = (1 - \gamma_{d-1})(\beta_d = 1) + \gamma_{d-1}(\beta_d \neq 1). \tag{3}$$

This can be seen if one bears in mind that γ represents just the carry in binary summation of σ_β to obtain σ_α.

Knowing the optimal path(s) from Lemma 1.1, we can also determine the best first move(s) (or edge(s)) to get from $\underline{s}is$ to $\underline{s}jt$ (cf. [8, p. 103]) with the aid of the following lemma.

Lemma 1.2. *Let $\underline{s}\bar{s} := \underline{s}is$ and j be as in Lemma 1.1. Then the optimal first move from $\underline{s}\bar{s}$ to $\underline{s}ji^n$ in S_p^N is from $\underline{s}\bar{s}_\delta j^{\delta-1}$ to $\underline{s}j\bar{s}_\delta^{\delta-1}$, where $\delta \in [n+1]$, $\bar{s}_\delta \neq j$ and $\underline{s} = \underline{s}\bar{s}_{n+1} \dots \bar{s}_{\delta+1} \in [p]_0^{N-\delta}$ (i.e. $\delta = \min\{d \in [n+1] \mid \bar{s}_d \neq j\}$).*

Proof. Again we may ignore $\underline{s}$. According to Lemma 1.1, the optimal path from $\bar{s}$ to ji^n is unique and consequently so is its first edge. In the special case $s = j^n$, this first edge is obviously directly to the goal, i.e. $\delta = n+1$. In all other cases, the first move is the same as for $s \to j^n$ with disc $n+1$ fixed on peg i. By definition of $E(S_p^n)$ it has to be of the form $\{s, t\}$ with $s = \underline{s}gh^{d-1}$ and $t = \underline{s}hg^{d-1}$ where $\{g, h\} \in \binom{[p]_0}{2}$, $d \in [n]$, and $\underline{s} \in [p]_0^{n-d}$ (not the original one!). Moreover, $\mathrm{d}(s, j^n) = \mathrm{d}(t, j^n) + 1$, such that from (0) we obtain

$$(g \neq j) \cdot 2^{d-1} + (h \neq j)(2^{d-1} - 1) = (h \neq j) \cdot 2^{d-1} + (g \neq j)(2^{d-1} - 1) + 1,$$

which is equivalent to $(g \neq j) - (h \neq j) = 1$. This, in turn, can only be true if $g \neq j = h$. □

Remark 1.3. The proof shows that in the Switching Tower of Hanoi interpretation of the task $j^n \neq s \to j^n$, $s \in [p]_0^n$, the best first move is to switch the smallest disc not on the goal peg with the tower of smaller ones on that peg. A peg different from j and empty in s is not used for the optimal solution. As a consequence, the optimal path from 0^n to 1^n in *any* graph S_p^n, $p \geq 2$, can be obtained by successive binary addition of 1 since all states on it are "binary"; cf. [8, p. 103].

The reader is again invited to test Lemma 1.2 on the minimal example $01 \to 21$ in S_3^2 for both optimal solutions, putting either $j = 2$ or $j = 1$.

2. The algorithm

In what follows p will be a fixed positive natural number. The input for our algorithm is a positive integer N and vertices $s, t \in [p]_0^N$. (Recall that there are p pegs, N discs and p^{2N} such pairs.) These data will not be altered by the procedures. The states s and t are entered as the sequence of pairs (s_d, t_d) downwards starting from $d = N$. We begin with a pre-processing procedure pre(N, s, t) eliminating those initial d for which $s_d = t_d$, i.e. omitting $\underline{s}$ according to Lemma 1.1; in particular, this will, if ending with $d = 0$, filter out the trivial case $s = t$ (including $p = 1$ or $N = 0$) for which we stop with "NO MOVE". (This is necessary, because assuming that $s \neq t$ may amount to screen all inputs!) For the remaining cases, the largest disc to be moved d is identified. If $d = 1$, we can immediately go to post-processing post$(0, s, t, p, 1)$ allocating the aspired data (cf. infra). Also if $p = 2$, we move to post$(d-1, s, t, 2, 1)$. Otherwise, the input data will be transferred to an initializing part of the algorithm making use, according to Proposition 1.0, of the first two pairs with $d = n+1$ and $d = n$. Apart from the trivial cases already sorted out (these are p^{N+1} for $p \geq 3$ and $N \geq 1$), all inputs s, t have to run through this screening realized in our Automaton 0 (see Fig. 2),[4] a fact that was overlooked in the statistical analysis for the case $p = 3$ in [13], where it was claimed that the expected value for the number of pairs read is $\frac{63}{38}$, i.e. less than 2; see the discussion in [6, p. 147]. For some of the inputs, and in particular if $n = 1$, the decision about the number of LDMs is already accomplished now (state D of the automaton) and we may directly proceed to post-processing. For some of the remaining cases, the third (and further) input pair(s), namely (starting with) $d = n-1$, will be entered in an automaton almost identical to Romik's (cf. [6, Figure 2.27]), our Automaton 2, with transient states A, B, and C and absorbing states D and E (see Fig. 4), based on Proposition 1.2. If, however, the four entries from the first two pairs are all different, we have to run through an intermediate Automaton 1 (see Fig. 3) which will, according to Proposition 1.1, either move on to post-processing or specify the unique shortcut peg k and pass to Automaton 2. Note that this can, of course, only happen if $p \geq 4$ and therefore this is the main feature of our algorithm that is essentially different from Romik's original approach applying only for $p = 3$, where there was no choice but $k = 3 - s_{n+1} - t_{n+1}$.

With the fixed data $n \in [N]_0$, $s, t \in [p]_0^N$, $k \in [p+1]_0 \setminus \{s_{n+1}, t_{n+1}\}$, and $b \in \{0, 1, 2\}$ we perform post-processing post(n, s, t, k, b) based on formulas (1), (2), and (3), and on Lemma 1.2 in an obvious way. Note that $k = p$ is a dummy if there is no bypass peg (to be) specified.

In a condensed form the algorithm is presented in Algorithm 1.

2.1. Some applications

Algorithm 1 can be employed to recover or extend known results on metric properties of Sierpiński graphs. For instance, for the so-called *P1 tasks* (starting or) ending in an extreme vertex/perfect state we have [8, Lemma 4]:

Proposition 2.1. *Minimal paths starting or ending in an extreme vertex are unique. For non-trivial tasks of that kind the LD moves exactly once, i.e.* $b = 1$.

Proof. By symmetry and w.l.o.g. we may assume that the task is (ii^n, jt) with $\{i, j\} \in \binom{[p]_0}{2}$, $t \in [p]_0^n$, and $n \in \mathbb{N}$. Since the first entry of every input pair is i, the process ends either in D or in A. □

On the other hand, extreme vertices are the only ones with this property; more precisely, we have the following result (cf. [5, Corollary 3.6] or [6, Proposition 4.2] for $p = 3$ and [15, Corollary 3.5] for a weaker version for general $p \geq 3$).

[4] We use the word "automaton" in a non-technical fashion; actually our three automata are the constituent parts of a single *finite-state automaton*.

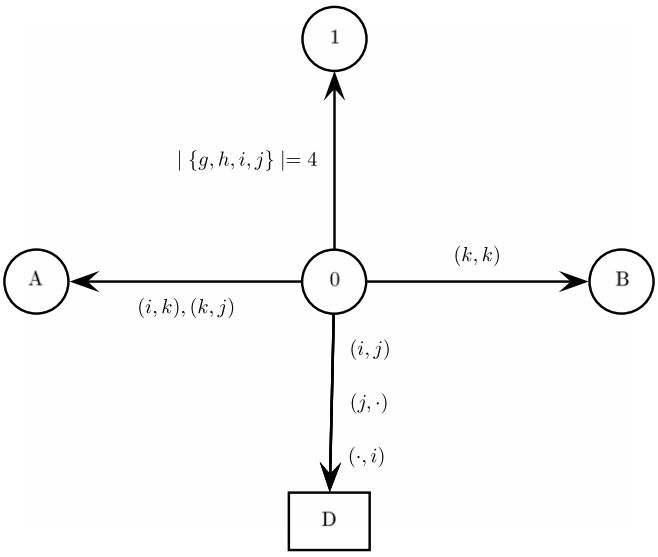

Fig. 2. Automaton 0. The input pairs at the arrows are (g, h); a dot stands for an arbitrary entry and $k \in [p]_0 \setminus \{i, j\}$.

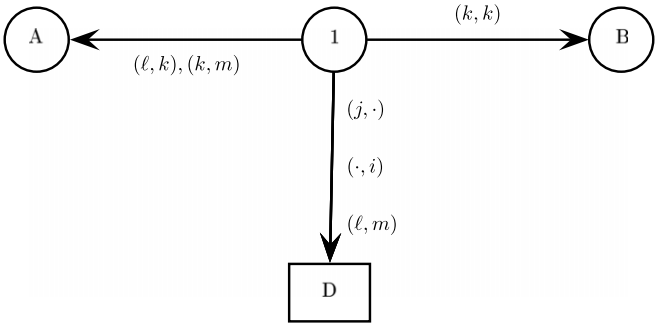

Fig. 3. Automaton 1. $\ell \in [p]_0 \setminus \{g, h, j\}$, $m \in [p]_0 \setminus \{g, h, i\}$ and $k \in \{g, h\}$. Note that for the remaining cases of input, namely (g, h) and (h, g), the automaton stays in state 1 and takes in the next input pair, if any.

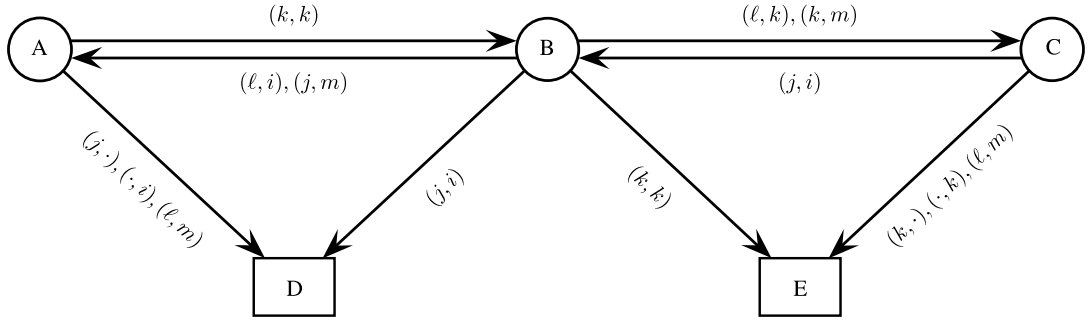

Fig. 4. Automaton 2. $\ell \in [p]_0 \setminus \{j, k\}$ and $m \in [p]_0 \setminus \{i, k\}$. For the remaining cases of input, namely (ℓ, k) or (k, m) in A, (j, k), (k, i) or (ℓ, m) in B and (ℓ, i) or (j, m) in C, the automaton stays in the respective state and takes in the next input pair, if any.

Proposition 2.2. *Let $p \geq 3$. For all non-extreme vertices $s \in [p]_0^N$ there is a $t \in [p]_0^N$ such that there are two shortest s, t-paths, i.e. $b = 0$. The vertex t can be chosen in such a way that $s_N \neq t_N$, i.e. the LD is N.*

Proof. Since s is non-extreme, there is some $\{i, k\} \in \binom{[p]_0}{2}$, $\delta \in [N-1]$ and $\bar{s} \in [p]_0^{\delta-1}$ such that $s = i^{N-\delta}k\bar{s}$. Choose $j \in [p]_0 \setminus \{i, k\}$ (this is where $p \geq 3$ enters; it is clear that the conclusion of the proposition cannot be true for $p = 2$) and put $t = jk^{N-\delta}\bar{t}$ with, for $d \in [\delta - 1]$,

$$t_d = k, \quad \text{if } s_d = j; \qquad t_d = i, \quad \text{if } s_d = k; \qquad t_d = j \quad \text{otherwise.}$$

Then the first $N - \delta + 1$ input pairs are (i, j), (i, k) $(N - \delta - 1$ times), (k, k), such that we land in state B. The remaining $\delta - 1$ input pairs are of the form (j, k), (k, i) or (ℓ, j) with $\ell \in [p]_0 \setminus \{j, k\}$, such that state B will not be left anymore. □

Algorithm 1 The shortest path algorithm for S_p^N ($p \in \mathbb{N}$ fixed)

Procedure shortestpath(N, s, t)
Parameter N: number of discs $\{N \in \mathbb{N}_0\}$
Parameter s: initial state $\{s \in [p]_0^N\}$
Parameter t: final state $\{t \in [p]_0^N\}$
pre(N, s, t) {out: NO MOVE if $d = 0$ or LD if $d \in [N]$; post(0, s, t, p, 1) if $d = 1$, post($d-1$, s, t, 2, 1) if $p = 2$}
$n \leftarrow d - 1$ $\{n \in [N-1]\}$
$(i, j) \leftarrow (s_d, t_d), (g, h) \leftarrow (s_n, t_n)$ {first two input pairs; $\{i, j\} \in \binom{[p]_0}{2}$, $g, h \in [p]_0$}
apply Automaton 0
  if Automaton 0 ends in state D **then** post(n, s, t, p, 1) **end if**
  if Automaton 0 ends in state 1 **then**
    apply Automaton 1, starting in state 1
      if Automaton 1 ends in state D or 1 **then** post(n, s, t, p, 1) **end if**
      if Automaton 1 ends in state A **then** apply Automaton 2, starting in state A
        else apply Automaton 2, starting in state B **end if**
  end if
  if Automaton 0 ends in state A **then** apply Automaton 2, starting in state A
    else apply Automaton 2 , starting in state B **end if**
if Automaton 2 ends in state D or A **then** post(n, s, t, p, 1) **end if**
if Automaton 2 ends in state E or C **then** post(n, s, t, k, 2) **end if**
post(n, s, t, k, 0)

In [10], S. Klavžar and S.S. Zemljič introduced the notion of *almost-extreme* vertices for the immediate neighbours of extreme vertices. They are examples of start- or goal-vertices for which all tasks need less than 2 LDMs; cf. [10, Proposition 4]:

Proposition 2.3. *For all tasks* $(i^{\nu+n}\kappa, i^\nu jt)$ *(or* $(i^\nu jt, i^{\nu+n}\kappa)$*) with* $\{i, j\} \in \binom{[p]_0}{2}$, $\kappa \in [p]_0 \setminus \{i\}$, $t \in [p]_0^n$, $\nu \in \mathbb{N}_0$, *and* $n \in \mathbb{N}$ *we have* $b < 2$; $b = 0$ *iff* $\kappa \neq j$ *and* $t = \kappa^n$.

Proof. By pre-processing we may assume that $\nu = 0$. Then the input sequence is (i, j), (i, t_d) for d from n down to 2, and finally (κ, t_1).

If $n = 1$, we end up in D, 1 or A, except when $t_1 = \kappa \neq j$, in which case we stop in B.

For $n > 1$, Automaton 0 leads directly to D if $t_n \in \{i, j\}$ or fixes $k = t_n$ and passes on to A. In the latter case, the rest of the input can only lead to B in the last step (and therefore never to C or E) and only if $t_d = k = \kappa$ for all remaining $d \in [n-1]$. □

For the last statement in Proposition 2.3, cf. [15, Theorem 3.3].

The result of [10, Proposition 7] can be obtained in a similar way from Algorithm 1. Moreover, another application of the algorithm shows that for every almost-extreme vertex of the form ik^n, $\{i, k\} \in \binom{[p]_0}{2}$, $p \geq 3$, $n \in \mathbb{N}$, i.e. incident with an edge corresponding to a transfer of the largest disc, the set of those vertices to (or from) which two shortest paths lead is composed of $\underline{t}ki^{\delta-1}$, where $\delta \in [n]$ and $\underline{t} \in ([p]_0 \setminus \{i, k\})^{n-\delta+1}$; for this result cf. [15, Theorem 3.1].

3. The statistics

In pre-processing we get $d = 0$ for p^N trivial tasks (s, s) where no move is performed, but all N input pairs (s_ν, t_ν) have been checked. The LD is $d \in [N]$ for $(p-1)p^{N+d-1}$ tasks of the form $(\underline{s}is, \underline{s}jt)$ where $\{i, j\} \in \binom{[p]_0}{2}$, $\underline{s} \in [p]_0^{N-d}$, and $s, t \in [p]_0^{d-1}$; for these, $N-d+1$ input pairs have been examined. The average number of input pairs used in pre-processing is therefore

$$p^{-2N}\left\{p^N N + \sum_{d=1}^{N}(p-1)p^{N+d-1}(N-d+1)\right\} = p^{-N}\left\{N + (p-1)\sum_{n=0}^{N-1}(N-n)p^n\right\}$$

$$= p^{-N}\left\{N + (p-1)^{-1}\left(p(p^N-1) - N(p-1)\right)\right\} = \frac{p^N-1}{(p-1)p^{N-1}} < \frac{p}{p-1}.$$

Moreover, for $N \geq 1$ and $d = 1$, all $(p-1)p^N$ non-trivial tasks result in $b = 1$. So pre-processing filters out already p^{N+1} from overall p^{2N} tasks. Therefore we may assume for our further statistical analysis that $p \geq 3$ and $N \geq 2$. The first step is then to apply Automaton 0 to the remaining $p^{N+1}(p^{N-1}-1)$ tasks. For each of these tasks this takes in 2 pairs. From the p^2 possible combinations of the second pair, $2p$ will lead, depending on the first pair, directly to $b = 1$, $2(p-2)$ will be passed on to state A and $p-2$ to state B of Automaton 2, and the remaining $(p-3)(p-2)$ will go to state 1, i.e. to Automaton 1. Note that no input in Automaton 0 will lead directly to states C or E of Automaton 2.

3.1. Total number of tasks with two optimal solutions

After the application of Automaton 0 already a total of $p^N(2p^{N-1}+p-3)$ non-trivial tasks have been assigned $b=1$. (Note that for the classical case $p=3$ these are $\frac{2}{3}$ of all tasks!) There are now $(p-2)p^N(p^{N-1}-1)$ tasks to be processed further in Automata 1 and 2. Of these, $(p-3)(p-2)p^{N-1}(p^{N-1}-1)$ start in state 1, $2(p-2)p^{N-1}(p^{N-1}-1)$ in state A, $(p-2)p^{N-1}(p^{N-1}-1)$ in state B and none in C, D or E. One may now, of course, be interested in finding the values of

$$B_\tau(N)=\left|\left\{(s,t)\in[p]_0^{2N}\mid b(s,t)=\tau\right\}\right|,$$

where $b(s,t)$ is the value of b assigned to task (s,t) (if $s\neq t$) and $\tau\in\{0,1,2\}$. In particular, we will focus on the question how many tasks have two optimal solutions, i.e. we want to determine B_0. Again we assume that $p\geq 3$ is fixed.

The adjacency matrix of the weighted directed graph for the non-terminal states 1, A, B, and C is (cf. [5, p. 703], where state 1 is obsolete)

$$M:=\begin{bmatrix}2 & 4(p-3) & 2 & 0\\ 0 & 2(p-2) & 1 & 0\\ 0 & 2(p-2) & (p-2)^2+2 & 2(p-2)\\ 0 & 0 & 1 & 2(p-2)\end{bmatrix}.$$

Its eigenvalues are 2, $2(p-2)$, and $\Theta_\pm:=\frac{1}{2}\left((p-1)^2+1\pm\sqrt{\alpha}\right)$, where $\alpha:=p^4-12p^3+56p^2-104p+68$. The matrix M can be diagonalized as $M=UDU^{-1}$ with

$$D:=\begin{bmatrix}2 & 0 & 0 & 0\\ 0 & 2(p-2) & 0 & 0\\ 0 & 0 & \Theta_+ & 0\\ 0 & 0 & 0 & \Theta_-\end{bmatrix}$$

and

$$U:=\begin{bmatrix}1 & 2 & 2 & 2\\ 0 & 1 & 1 & 1\\ 0 & 0 & \Xi_+ & \Xi_-\\ 0 & -1 & 1 & 1\end{bmatrix},\qquad U^{-1}=\frac{1}{2\sqrt{\alpha}}\begin{bmatrix}2\sqrt{\alpha} & -4\sqrt{\alpha} & 0 & 0\\ 0 & \sqrt{\alpha} & 0 & -\sqrt{\alpha}\\ 0 & -\Xi_- & 2 & -\Xi_-\\ 0 & \Xi_+ & -2 & \Xi_+\end{bmatrix},$$

where $\Xi_\pm:=\Theta_\pm-2(p-2)=\frac{1}{2}\left((p-3)^2+1\pm\sqrt{\alpha}\right)$, such that for $n\in\mathbb{N}_0$: $M^n=UD^nU^{-1}$. To calculate B_0, we determine $M^n_{13}=\frac{2}{\sqrt{\alpha}}(\Theta_+^n-\Theta_-^n)$, $M^n_{23}=\frac{1}{\sqrt{\alpha}}(\Theta_+^n-\Theta_-^n)$, and $M^n_{33}=\frac{1}{\sqrt{\alpha}}(\Xi_+\Theta_+^n-\Xi_-\Theta_-^n)$; then

$$\begin{aligned}B_0(N)&=(p-2)(p-1)p^{N-1}\sum_{n=0}^{N-2}p^{-n}\left\{(p-3)M^n_{13}+2M^n_{23}+M^n_{33}\right\}\\&=(p-2)(p-1)p\sum_{n=1}^{N-1}p^{N-1-n}M^n_{23}\\&=\frac{(p-2)(p-1)p}{\sqrt{\alpha}}\sum_{n=0}^{N-1}p^{N-1-n}(\Theta_+^n-\Theta_-^n),\end{aligned}$$

where we have made use of $M^n_{13}=2M^n_{23}$ and $M^n_{33}=M^{n+1}_{23}-2(p-2)M^n_{23}$. Employing the telescoping formula

$$(x-y)\sum_{n=0}^{N-1}x^{N-1-n}y^n=x^N-y^N,$$

we arrive at

Proposition 3.1. *For $p\in\mathbb{N}$ and $N\in\mathbb{N}_0$,*

$$B_0(N)=\frac{(p-1)p}{p^2-7p+8}\left\{p^N+\frac{\Theta_--p}{\sqrt{\alpha}}\Theta_+^N-\frac{\Theta_+-p}{\sqrt{\alpha}}\Theta_-^N\right\}.$$

For $p=3$, this is [6, Proposition 2.39]. For $p=4$, we have $\alpha=36$, $\Theta_+=8$, $\Theta_-=2$, whence in this case

$$B_0(N)=2^N(4^N-3\cdot 2^N+2).$$

The method can, of course, be applied to find B_2, and (consequently) B_1, by considering the full adjacency matrix including the absorbing states D and E. The latter two contribute the extra (double) eigenvalue p^2. However, the calculations get a bit clumsy, and there is an alternative method based on the utmost interesting counting functions, given for fixed $p\geq 2$ by (cf. [6, (2.18)]):

$$\forall\, n\in\mathbb{N}_0\ \forall\,\mu\in\mathbb{Z}:\ z_n(\mu)=|\{s\in[p]_0^n\mid \mathrm{d}(s,1^n)-\mathrm{d}(s,0^n)=\mu\}|.$$

They have been introduced and intensely studied for the standard case $p = 3$ in [3, p. 305ff] and discussed in connection with *Stern's diatomic sequence* in [11] and [5, p. 700ff]. It turned out that their properties extend to all p (cf. [14, p. 27ff]) and can, e.g., be used as in [3] to deduce that among the $(p-1)p^{N+n}$ tasks where the LD is $n+1$ ($n \in [N]_0$)

$$\frac{(p-2)(p-1)p^{N-n}}{2}\left\{\frac{p^{2n}-2^n(p-2)^n}{p^2-2p+4}-\frac{\Theta_+^n-\Theta_-^n}{\sqrt{\alpha}}\right\}$$

lead to $b = 2$. For $p = 3$ (and $n = N-1$), this is [3, Proposition 6ii] (cf. [6, Proposition 2.40]); for general p it corresponds to [14, Satz 3.1.8ii]. For $p = 4$, the expression is particularly simple, namely $4^{N-1-n} \cdot 2^n(8^n - 2 \cdot 4^n - 2^n + 2)$, such that in this case

$$B_2(N) = 2^N\left(\frac{1}{12}\,8^N - \frac{1}{2}\,4^N + \frac{17-3N}{12}\,2^N - 1\right)$$

and

$$B_1(N) = 2^N\left(\frac{11}{12}\,8^N - \frac{1}{2}\,4^N + \frac{7+3N}{12}\,2^N - 1\right).$$

One may wonder whether, apart from the obvious fact that α is a square for $p = 4$, there is a combinatorial explanation for this case to stand out for simplicity.

3.2. Complexity of the decision algorithm

In [13, Section 5] an analysis has been performed on the expected number of input pairs needed for the P2 decision problem based on a Markov process (cf. also [6, p. 146ff]). In our more general setting and if we start again from the situation *after* Automaton 0 has been passed, i.e. with the states 1, A, B, C, D, and E, the corresponding transition matrix is $\begin{bmatrix} Q & R \\ 0 & I^2 \end{bmatrix}$, where $Q = p^{-2}M$, $I^2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, and

$$R = p^{-2}\begin{bmatrix} (p-3)^2+2p-1 & 0 \\ (p-2)^2+2p-1 & 0 \\ 1 & 1 \\ 0 & (p-2)^2+2p-1 \end{bmatrix}.$$

Our process is therefore an absorbing Markov chain, such that $Q^n \to 0$ as $n \to \infty$ and $I^4 - Q$ has the inverse $N := (I^4-Q)^{-1} = \sum_{n=0}^{\infty} Q^n$, called the *fundamental matrix* of the process. Its entry N_{uv} is the expected number of times the chain is in state v provided it had started in state u. We have

$$N = p^2\begin{bmatrix} p^2-2 & -4(p-3) & -2 & 0 \\ 0 & (p-1)^2+2 & -1 & 0 \\ 0 & -2(p-2) & 2(2p-3) & -2(p-2) \\ 0 & 0 & -1 & (p-1)^2+2 \end{bmatrix}^{-1}.$$

To obtain the expected number of times X_p we will be in transient states is then (recall that there is zero a-priori probability to start the process in state C)

$$p^{-2}\left\{(p-3)(p-2)\sum_{v=1}^{4}N_{1v} + 2(p-2)\sum_{v=1}^{4}N_{2v} + (p-2)\sum_{v=1}^{4}N_{3v}\right\},$$

which amounts to

$$X_p = \frac{5p^5-8p^4-72p^3+272p^2-352p+160}{2(2p^5-7p^4+8p^3+6p^2-24p+16)}.$$

We therefore arrive at

Theorem 3.1. *The average number of input pairs checked by Algorithm* 1 *for tasks in* S_p^N after *pre-processing, when the LD is* $n+1 \in [N]$, *is bounded above by and converges, as* $n \to \infty$, *to* $2 + X_p$.

Remark 3.1. In all three automata there are cases where already the first (or either the last) entry of an input pair decides about the outcome. This reduction in input data can be dealt with as before. In order to preserve comparability with previous results, we do not go into this here; cf. the discussion in [6, p. 147f].

The first few values for X_p are $X_3 = 25/38$, $X_4 = 1$, $X_5 = 6825/5842$, and $X_6 = 1300/1037$. We notice that the case $p = 3$ is in accordance with [13, Theorem 3] (cf. supra), while $p = 4$ again seems to be a bit special!

Acknowledgements

We thank Sara Sabrina Zemljič (Ljubljana) for a valuable discussion. We are also grateful to two anonymous reviewers for their constructive suggestions for improvement.

References

[1] S. Aumann, K.A.M. Götz, A.M. Hinz, C. Petr, Moves of the largest disc in shortest paths on Hanoi graphs (2014) submitted for publication.
[2] L.L. Cristea, B. Steinsky, Distances in Sierpiński graphs and on the Sierpiński gasket, Aequationes Math. 85 (2013) 201–219.
[3] A.M. Hinz, The Tower of Hanoi, Enseign. Math. (2) 35 (1989) 289–321.
[4] A.M. Hinz, Shortest paths between regular states of the Tower of Hanoi, Inform. Sci. 63 (1992) 173–181.
[5] A.M. Hinz, S. Klavžar, U. Milutinović, D. Parisse, C. Petr, Metric properties of the Tower of Hanoi graphs and Stern's diatomic sequence, European J. Combin. 26 (2005) 693–708.
[6] A.M. Hinz, S. Klavžar, U. Milutinović, C. Petr, The Tower of Hanoi—Myths and Maths, Springer, Basel, 2013.
[7] A.M. Hinz, D. Parisse, The average eccentricity of Sierpiński graphs, Graphs Combin. 28 (2012) 671–686.
[8] S. Klavžar, U. Milutinović, Graphs $S(n, k)$ and a variant of the Tower of Hanoi problem, Czechoslovak Math. J. 47 (122) (1997) 95–104.
[9] S. Klavžar, I. Peterin, S.S. Zemljič, Hamming dimension of a graph—The case of Sierpiński graphs, European J. Combin. 34 (2013) 460–473.
[10] S. Klavžar, S.S. Zemljič, On distances in Sierpiński graphs: almost-extreme vertices and metric dimension, Appl. Anal. Discrete Math. 7 (2013) 72–82.
[11] D. Parisse, The Tower of Hanoi and the Stern–Brocot Array (Ph.D. thesis), Ludwig-Maximilians-Universität, München, 1997.
[12] D. Parisse, On some metric properties of the Sierpiński graphs $S(n, k)$, Ars Combin. 90 (2009) 145–160.
[13] D. Romik, Shortest paths in the Tower of Hanoi graph and finite automata, SIAM J. Discrete Math. 20 (2006) 610–622.
[14] H.-L. Wiesenberger, Stochastische Eigenschaften von Hanoi- und Sierpiński-Graphen (Diploma thesis), Ludwig-Maximilians-Universität, München, 2010.
[15] B. Xue, L. Zuo, G. Wang, G. Li, Shortest paths in Sierpiński graphs, Discrete Appl. Math. 162 (2014) 314–321.