MATHEMATICS OF OPERATIONS RESEARCH
Vol. 38, No. 3, August 2013, pp. 526–534
ISSN 0364-765X (print) | ISSN 1526-5471 (online)

http://dx.doi.org/10.1287/moor.1120.0582
© 2013 INFORMS

Learning Cycle Length Through Finite Automata

Ron Peretz
Department of Mathematics, London School of Economics, London WC2A 2AE, United Kingdom, ronprtz@gmail.com,
http://www2.lse.ac.uk/researchAndExpertise/Experts/profile.aspx?KeyValue=r.perez@lse.ac.uk

We study the space-and-time automaton-complexity of two related problems concerning the cycle length of a periodic stream of input bits. One problem is to find the exact cycle length of a periodic stream of input bits provided that the cycle length is bounded by a known parameter n. The other problem is to find a large number k that divides the cycle length. By "large" we mean that there is an unbounded increasing function $f(n)$, such that either k is greater than $f(n)$ or k is the exact cycle length.

Our main results include that finding a large divisor of the cycle length can be solved in deterministic linear TIME and sub-linear SPACE, whereas finding the exact cycle length cannot be solved in deterministic TIME × SPACE smaller than a constant times n squared. Results involving probabilistic automata and applications to rate-distortion theory and repeated games are also discussed.

Key words: automaton-complexity; games with bounded complexity; rate-distortion theory; sub-linear space algorithm
MSC2000 subject classification: Primary: 68Q45; secondary: 91A26, 68Q30
OR/MS subject classification: Primary: statistics: pattern analysis; secondary: games/group decisions: noncooperative; analysis of algorithms: computational complexity
History: Received May 11, 2010; revised February 23, 2011, December 13, 2011, November 8, 2012. Published online in *Articles in Advance* March 13, 2013.

1. Introduction. We study two related problems, CYCLE-LENGTH and CYCLE-DIVISOR. The input of these problems is a periodic stream of bits whose cycle length is bounded by a known parameter n. In the CYCLE-LENGTH problem the output is the exact cycle length. In the CYCLE-DIVISOR problem, the output is either the exact cycle length or a large divisor of the cycle length, a number greater than some function of n that diverges to infinity as n grows. The complexity is measured in terms of the SPACE, the logarithm of the number of states in an automaton that solves the problem, and the TIME required to reach a terminal state. We also consider the SPACE × TIME complexity which is the minimum of the function SPACE · TIME over all finite automata that solve the problem. We analyze the worst input against a deterministic automaton, and against a probabilistic automaton (a probability measure over deterministic automata). In the probabilistic case we require that the probability of computing a correct output is arbitrarily close to one.

Our findings can be summarized as follows:

- CYCLE-DIVISOR can be solved in deterministic SPACE $o(n)$, and TIME $O(n)$.
- CYCLE-LENGTH cannot be solved in deterministic SPACE × TIME smaller than $\Omega(n^2)$.
- CYCLE-LENGTH can be solved in probabilistic SPACE $o(n)$, and TIME $O(n)$.
- CYCLE-LENGTH can be solved in deterministic SPACE $O(nL)$, and TIME $O(n/L)$, for any positive $L \leq 1$.

The above says that CYCLE-DIVISOR is strictly easier than CYCLE-LENGTH. In fact, our positive results are all reductions to the CYCLE-DIVISOR problem. Our first theorem provides an upper bound for the deterministic complexity of CYCLE-DIVISOR. We do not know if this bound is tight. We are also unaware of a better upper bound for probabilistic CYCLE-DIVISOR.

Section 4 contains an application of the CYCLE-DIVISOR upper bound to the automaton-complexity of minimal distortion functions, a topic in information theory. In §5 we discuss the motivation behind this work, repeated games with finite automata.

2. Results. A deterministic finite automaton is a tuple $\langle \Sigma, S, s_*, f, H, O, g \rangle$, where

- Σ is a finite set of two or more elements, the input alphabet;
- S is a finite set, the states;
- $s_* \in S$ is the initial state;
- $f\colon S \times \Sigma \to S$ is the transition function;
- $H \subset S$ is the set of terminal states;
- O is the output domain;
- $g\colon H \to O$ is the output function.

Given a sequence of input letters $a_1, a_2, \ldots$, the *run* of the automaton is a sequence of states $s_1, s_2, \ldots$, defined recursively by

$$s_1 = s_*, \qquad s_{t+1} = f(s_t, a_t).$$

We say that an automaton halts at time t given the input a, if t is the first time the run visits a terminal state. That is, $t=\min\{t'\colon s_{t'}\in H\}$. In this case, we say that, given a, the automaton halts in t steps outputting $g(s_t)$.

Let Σ be a finite alphabet. The set of n-periodic sequences is denoted $\Sigma^{(n)}=\{(a_t)\in\Sigma^{\mathbb{N}}\colon \forall t\in\mathbb{N}\ a_t=a_{t+n}\}$. The set of periodic sequences whose cycle length is at most n is denoted $\Sigma^{(\le n)}=\bigcup_{k=1}^{n}\Sigma^{(k)}$. The exact cycle length of a periodic sequence a, denoted $\rho(a)$, is the smallest integer n such that a is n-periodic. Formally, for $a\in\bigcup_{k=1}^{\infty}\Sigma^{(k)}$, $\rho(a)=\min\{k\colon a\in\Sigma^{(k)}\}=\gcd\{k\colon a\in\Sigma^{(k)}\}$.

We refer to the cardinality of the input alphabet $|\Sigma|$ as a constant number in our asymptotic analysis.

Our first theorem provides an upper bound for the complexity of the CYCLE-DIVISOR problem. This is the main result. Theorems 3, 4 and 6, as well as the solution to Neyman's problem in repeated games, involve the use of Theorem 1.

THEOREM 1. *There exists a deterministic finite automaton with* $2^{O(\sqrt{n\log n})}$ *states such that, for any input* $a\in\Sigma^{(\le n)}$, *the automaton halts in* $2(n+\sqrt{n\log n})$ *steps outputting a number* k *that divides* $\rho(a)$, *and if* $k<\sqrt{n\log n}$, *then* $k=\rho(a)$.

The next theorem shows that the CYCLE-LENGTH problem is strictly harder than the CYCLE-DIVISOR problem.

THEOREM 2. *The deterministic TIME × SPACE complexity of CYCLE-LENGTH is* $\Omega(n^2)$.

Randomization, however, can speed up the solution. For a fixed size m, a probabilistic finite automaton with m states is a random variable that assumes values in the class of deterministic finite automata of m states.

THEOREM 3. *There exists a probabilistic finite automaton with* $2^{O(\sqrt{n\log n})}$ *states that finds the exact cycle length in* $(4+o(1))n$ *steps with probability greater than* $1-1/n$.

Finally, we show that the lower bound provided by Theorem 2 is tight up to a constant factor.

THEOREM 4. *For every* $0<L\le 1$ *there exists a deterministic finite automaton with* $2^{O(nL)}$ *states that finds the exact cycle length in* $O(n/L)$ *steps.*

3. Proofs.

We begin with two simple observations that refer to an *arbitrary* finite alphabet Σ.

CLAIM 1. *The number of elements in* $\Sigma^{(\le n)}$ *is less than* $2|\Sigma|^n$.

PROOF.

$$|\Sigma^{(\le n)}|=\left|\bigcup_{k=1}^{n}\Sigma^{(k)}\right|\le\sum_{k=1}^{n}|\Sigma^{(k)}|=\sum_{k=1}^{n}|\Sigma|^k=\frac{|\Sigma|}{|\Sigma|-1}(|\Sigma|^n-1).\quad\square$$

CLAIM 2. *For any finite alphabet* Σ, *the map* $a\mapsto(a_1,\dots,a_{2n})$, *from* $\Sigma^{(\le n)}$ *to* Σ^{2n}, *is injective.*

PROOF. Suppose a,b are in $\Sigma^{(\le n)}$ and $(a_1,\dots,a_{2n})=(b_1,\dots,b_{2n})$. Let $1\le k\le n$. If $a\notin\Sigma^{(k)}$, then there exists $1\le i\le n$ such that $a_i\ne a_{i+k}$; so $b_i\ne b_{i+k}$; so $b\notin\Sigma^{(k)}$. Similarly, if $b\notin\Sigma^{(k)}$, then $a\notin\Sigma^{(k)}$; so $\rho(a)=\rho(b)$. Since $(a_1,\dots,a_{\rho(a)})=(b_1,\dots,b_{\rho(b)})$, $a=b$. $\square$

We are now ready to prove Theorem 1.

PROOF OF THEOREM 1. Let $m=[\sqrt{n\log n}]$. For every input sequence $a=(a_t)_{t=1}^{\infty}\in\Sigma^{(\le n)}$, we define a set of positive integers $T_0(a)$ by

$$T_0(a)=\{t\ge 2m\colon (a_{t-2m+1},\dots,a_t)\notin\mathbf{pref}\,\Sigma^{(\le m)}\},$$

where "$\mathbf{pref}\,X$" denotes the set of all finite prefixes of sequences in X. Let

$$t_0(a)=\begin{cases}\min T_0(a) & \text{if } T_0(a)\ne\varnothing,\\ \infty & \text{otherwise.}\end{cases}$$

We use T_0 and t_0 for $T_0(a)$ and $t_0(a)$ when it causes no confusion.

Note that $T_0(a)$ is a $\rho(a)$-periodic set. Namely, $t\in T_0(a)$ iff $t+\rho(a)\in T_0(a)$, for every $t\ge 2m$. Since $\rho(a)\le n$, we have either $t_0(a)=\infty$ or $t_0(a)<n+2m$. Note, too, that $t_0(a)$ is a stopping time, namely, the question of whether $t<t_0(a)$ can be answered by looking at $a_1,\dots,a_t$ or not.

We describe an automaton $\langle\Sigma,S,s_*,f,H,\mathbb{N},g\rangle$. The states are partitioned into two disjoint sets $S=S_1\dot\cup S_2$. The states in S_1 are visited during time $t<t_0$, and the states in S_2 are visited during time $t\ge t_0$. Given an input sequence a, we shall first describe the state visited at any time t, s_t, as a function of $(a_1,\dots,a_t)$, and then argue that s_t is, indeed, a function of s_{t-1} and a_t.

Before time t_0. Define the set of states S_1 as follows:

$$S_1=\{(a_1,\dots,a_t)\in\mathbf{pref}\,\Sigma^{(\le n)}\colon 0\le t\le 2n,\forall t'=2m,\dots,t\ (a_{t'-2m+1},\dots,a_{t'})\in\mathbf{pref}\,\Sigma^{(\le m)}\}.$$

Note that the set of states S_1 consists of all the possible histories that may occur before time $t_0(a)$ and no later than time $2n$, for any input a.

The initial state, s_*, is the empty history. The terminal states in S_1, $S_1 \cap H$, are the histories of length $2n$ in S_1, $S_1 \cap \Sigma^{2n}$. For $t < \min\{t_0, 2n+1\}$, $s_t = (a_1, \ldots, a_t)$ (here we use that t_0 is a stopping time). Obviously, s_t is a function of s_{t-1} and a_t. For $s \in S_1 \cap H$, $s = (a_1, \ldots, a_{2n}) \in \mathbf{pref}\,\Sigma^{(\leq n)}$. By Claim 2, $(a_1, \ldots, a_{2n})$ determines a, as $a \in \Sigma^{(\leq n)}$. We define $g(s) = \rho(a)$, and (for completeness only) $f(s) = s$.

It remains to claim that $|S_1| = 2^{O(\sqrt{n \log n})}$. We shall prove the following inequality:

$$|S_1| \leq 2n|\Sigma|^{2m} m^{2n/m}. \tag{1}$$

To see this, consider an element $(a_1, \ldots, a_t) \in S_1$. Let $l = [t/m]$. Let $k_1, \ldots, k_{l-1} \leq m$ be integers such that $(a_{m(i-1)+1}, \ldots, a_{m(i+1)}) \in \mathbf{pref}\,\Sigma^{(k_i)}$. Note that k_i together with $(a_1, \ldots, a_{mi})$ determines $(a_1, \ldots, a_{m(i+1)})$; therefore the entire sequence $(a_1, \ldots, a_t)$ is determined by the following data:

- t,
- $a_1, \ldots, a_m$,
- $a_{lm+1}, \ldots, a_t$,
- $k_1, \ldots, k_{l-1}$.

Counting the number of possible values for each data item concludes (1).

Time t_0 until $2n+2m$. The set of states visited during time $t \geq t_0$, S_2, is defined by

$$B = \left\{ (b_1, \ldots, b_l) \in \{0,1\}^{\leq 2n} \colon \sum_{t=mi+1}^{m(i+1)} b_t \leq 1, \text{ for every } 0 \leq i < [l/m] \right\},$$

$$S_2 = \Sigma^{2m} \times \Sigma^{2m} \times B.$$

A straightforward calculation shows that $|S_2| = 2^{O(\sqrt{n \log n})}$.

Assume $t_0 < \infty$. Recall that this means that $t_0 < n + 2m$. We would like to describe s_t, for $t \geq t_0$, as a function of the input sequence $(a_t)_{t=1}^{\infty}$. Consider the following stationary coding of the input sequence:

$$b_t = \begin{cases} 1 & \text{if } (a_{t-2m+1}, \ldots, a_t) = (a_{t_0-2m+1}, \ldots, a_{t_0}), \\ 0 & \text{otherwise.} \end{cases}$$

Note that[1] there are at least m "zeros" between any two "ones" in $(b_t)_{t=1}^{\infty}$; therefore $(b_{2m}, \ldots, b_{2n+2m-1}) \in B$.

For $t_0 \leq t < 2n + 2m$, s_t is defined by

$$\begin{aligned} s_t = \langle\, &(a_{t_0-2m+1}, \ldots, a_{t_0}), \\ &(a_{t-2m+1}, \ldots, a_t), \\ &(b_{2m}, \ldots, b_t) \rangle. \end{aligned}$$

Such a definition allows the automaton to compute the next bit, b_{t+1}, in the transition from time t to $t+1$. Since s_t is a function of $(a_1, \ldots, a_t)$, the transition from time $t_0 - 1$ to t_0 is also well defined.

As a stationary coding of $(a_t)_{t=1}^{\infty}$, $\rho(b)$ divides $\rho(a)$. Since $\rho(b) \leq n$ the entire sequence $(b_t)_{t=1}^{\infty}$ can be deduced from $b_{2m}, \ldots, b_{2n+2m-1}$. At time $2n+2m-1$ the automaton outputs $\rho(b)$. As mentioned, the sparseness of $(b_t)_{t=1}^{\infty}$ guarantees that $\rho(b) > m$. □

The proof of Theorem 2 relies on a diagonal argument, called the "fooling set," commonly used in the theory of communication complexity. See, for example, Kushilevitz and Nisan [3, p. 10]. The proof reduces the well-studied string equality problem to the CYCLE-LENGTH problem. Consequently, the well-known SPACE × TIME lower bound for string equality applies here. For completeness, we present a self-contained proof.

Proof of Theorem 2. Assume by negation that there exists an automaton with 2^S states that solves CYCLE-LENGTH in T steps, and that $S \cdot T < (1/16)n^2$. Choose a prime number $n/4 \leq p \leq n/2$. Consider inputs of the form $x, x, \ldots,$ where $x \in \Sigma^p$. Any such input yields a sequence of states of the automaton, $s_1, \ldots, s_{[T/p]}$, where s_j is the state of the automaton at time pj. By the pigeonhole principle, there must be two inputs $x \neq y$ that yield the same sequence of states $s_1, \ldots, s_{[T/p]}$; therefore the sequence $x, y, x, y, \ldots,$ also yields $s_1, s_2, \ldots$. This is a contradiction[2] since the cycle length of $x, x, \ldots,$ differs from the cycle length of $x, y, x, y, \ldots$. □

[1] A (finite) sequence can overlap with itself by a shift of l places iff it is in $\mathbf{pref}\,\Sigma^{(l)}$. This brilliant argument was suggested by Prof. Benjamin Weiss.

[2] W.l.o.g., the output is given by the state at time T since we may assume that once the automaton visits a terminal state it stays there forever.

In the next proof we use the Rabin-Karp (Karp and Rabin [2]) hash function. Although any other hash function could be applied here, the Rabin-Karp hash function has the advantage that it can be computed incrementally. This fact simplifies the proof significantly.

Proof of Theorem 3. Assume $\Sigma = \{0, 1, \ldots, |\Sigma| - 1\}$. Let $a \in \Sigma^{(\leq n)}$. Let $m = [\sqrt{n \log n}]$. Apply Theorem 1 to find a number k that divides $\rho(a)$, and if $k < m$, then $k = \rho(a)$. This can be done with $2^{O(\sqrt{n \log n})}$ states in $2(n+m)$ steps. If $k < m$, output k. If $k \geq m$, let p be a random prime number, chosen uniformly from the set of prime numbers between 2 and n^3. For $t = 1, 2, \ldots$, let $b_t = \sum_{l=1}^{k} a_{kt+l} |\Sigma|^{k-l}$. Let c_t be the element of $\{1, \ldots, p\}$ congruent to b_t modulo p.

The[3] mapping $(a_{kt+1}, \ldots, a_{k(t+1)}) \mapsto c_t$ can be computed incrementally according to the rule

$$c_t^i = |\Sigma| c_t^{i-1} + a_{kt+i} \bmod p,$$

where $c_t^0 = 0$ and $c_t = c_t^k$. Since c_t^i and c_t assume values in a set of at most n^3 elements and $[n/k] = O(\sqrt{n/\log n})$, the sequence $c_1, \ldots, c_{2[n/k]}$ can be learned with $2^{O(\sqrt{n \log n})}$ states in $2n$ steps.

The number b_t encodes $a_{kt+1}, \ldots, a_{k(t+1)}$; therefore $\rho(a) = k\rho(b)$. Obviously, $\rho(c) \mid \rho(b)$ and $\rho(b) \leq [n/k]$; therefore $\rho(c)$ can be deduced from $c_1, \ldots, c_{2[n/k]}$. The automaton outputs $k\rho(c)$.

In the event that $\forall\, t, s[b_t \neq b_s \rightarrow c_t \neq c_s]$, we also have $\rho(b) \mid \rho(c)$, and hence $\rho(c) = \rho(b)$, and $k\rho(c) = \rho(a)$. It remains to estimate the probability of this event. The prime numbers theorem[4] and the fact that any integer $x > 2$ has less than $\log(x)$ distinct prime divisors ensures that if $b_t \neq b_s$, then $\Pr(b_t = b_s \bmod p) = O(n^{-3}(\log n)^2)$. Since s and t range between 1 and $[n/k]$, $\Pr(\forall\, t, s[b_t \neq b_s \rightarrow c_t \neq c_s]) = 1 - O(n^{-2} \log n)$. □

Proof of Theorem 4. Let $a \in \Sigma^{(\leq n)}$ be an input sequence. Let $m = [\sqrt{n \log n}]$. Apply Theorem 1 to find a number k that divides $\rho(a)$. If $k < m$, output k. Let us assume that $k \geq m$ and describe, for each possible value of k, an automaton whose initial state is the state where the automaton of Theorem 1 halts.

For a set $A = \{\alpha_1, \ldots, \alpha_l\} \subset \{1, \ldots, k\}$, consider the Σ^l-valued sequence b^A, defined by $(b_t^A)_i = a_{kt+\alpha_i}$. Note that $k\rho(b^{\{1, \ldots, k\}}) = \rho(a)$ and for every $A, B \subset \{1, \ldots, k\}$,

$$\rho(b^{A \cup B}) = \operatorname{lcm}(\rho(b^A), \rho(b^B)).$$

Let $l = [kL]$. Choose $A_1, \ldots, A_{\lceil L^{-1} \rceil} \subset \{1, \ldots, k\}$, such that $\bigcup_i A_i = \{1, \ldots, k\}$ and $|A_i| = l$, for every i. Let $n' = k[n/k]$. In the first $2n'$ steps the automaton learns the sequence b^{A_1}. This can be done since $\rho(b^{A_1}) \leq n'/k$, and the number of states required is $2^{O(nL)}$. Assume that at time $2n'i$ the automaton has learned $\rho(b^{A_1 \cup \cdots \cup A_i})$. In the next $2n'$ steps it learns $b^{A_{i+1}}$ and computes $\operatorname{lcm}(\rho(b^{A_1 \cup \cdots \cup A_i}), \rho(b^{A_{i+1}})) = \rho(b^{A_1 \cup \cdots \cup A_{i+1}})$. In doing so, the automaton computes $\rho(a) = k\rho(b^{\{1, \ldots, k\}})$ after $2n'\lceil L^{-1} \rceil$ steps. □

4. Minimum distortion functions.

Minimum distortion functions play a role in rate-distortion theory, a branch of information theory (see Cover and Thomas [1, Ch. 10]). In this section we present an application of Theorem 1 to the implementation of minimum distortion functions through finite automata.

4.1. General framework.

We describe the general framework of minimum distortion functions. Let $\langle Y, d \rangle$ be a metric space, $f\colon Y \rightarrow Y$ a function, and X a finite subset of Y. The *distortion* of f on X is defined by

$$\delta_X(f) = \|d(x, f(x))_{x \in X}\|,$$

for some norm on $\mathbb{R}^X$ (which has to be specified). The *rate* of f on X is defined by

$$R_X(f) = \frac{\log |f(X)|}{\log |X|}.$$

For a fixed rate R, the infimum of $\delta_X(f)$ over all functions f whose rate on X is at most R defines the distortion-rate function.

Now let $\{X_n\}_{n=1}^{\infty}$ be a sequence of finite subsets of Y. The distortion-rate function for a fixed rate R related to this sequence, which with abuse of notation we denote by just $\delta(R)$, is defined by

$$\delta(R) = \liminf_{n \rightarrow \infty} \inf_{\substack{f\colon Y \rightarrow Y \\ R_{X_n}(f) \leq R}} \delta_{X_n}(f).$$

[3] This is the Rabin-Karp hash function. See Karp and Rabin [2].

[4] We only use the fact that the number of primes up to n is $\Omega(n/\log n)$.

We say that a class $\mathscr{C}$ of sequences of functions from Y to Y (equivalently, functions from $Y \times \mathbb{N}$ to Y) obtains the distortion bound on $\{X_n\}_{n=1}^{\infty}$ at rate $R > 0$, if there exist sequences of functions $\{f_n: Y \to Y\}_{n=1}^{\infty} \in \mathscr{C}$, such that $R_{X_n}(f_n) \le R$ and

$$\limsup_{n\to\infty} \delta_{X_n}(f_n) \le \delta(R).$$

4.2. Implementation through finite automata. We now turn to discuss a special case of the general framework. The metric space is the set of periodic sequences of bits $\Sigma^{(<\infty)} = \bigcup_{k=1}^{\infty} \Sigma^{(k)}$, $\Sigma = \{0, 1\}$, equipped with the Hamming metric

$$d(a, b) = \lim_{N\to\infty} \frac{1}{N} |\{1 \le n \le N: a_n \ne b_n\}|.$$

We consider two distortion variants defined with different norms. The "worst-case distortion" uses the $\|\cdot\|_\infty$ norm, and the "average distortion" uses the (normalized) $\|\cdot\|_1$ norm. Formally, for a function $f: \Sigma^{(<\infty)} \to \Sigma^{(<\infty)}$ and a finite set $X \subset \Sigma^{(<\infty)}$, we define the *worst-case distortion* of f on X by

$$\delta_X(f) = \max_{x\in X} d(x, f(x)),$$

and the *average distortion* of f on X by

$$D_X(f) = \frac{1}{|X|} \sum_{x\in X} d(x, f(x)).$$

We consider two sequences of finite sets $\{\Sigma^{(n)}\}_{n=1}^{\infty}$ and $\{\Sigma^{(\le n)}\}_{n=1}^{\infty}$. It is shown below that the distortion-rate function remains the same for any one of the considered sequences of finite sets and distortion variants. Neyman [4] has shown that the class of $\Sigma^{(n)}$ invariant functions that can be implemented through deterministic finite automata with 2^{Rn} states, halting in $\mathrm{O}(n)$ steps, obtains the distortion bound on $\{\Sigma^{(n)}\}_{n=1}^{\infty}$ at rate R, for every $R > 0$, with respect to worst-case distortion. It is also shown in Neyman [4] that, by enlarging the class of automata to those that halt in $n \log n/\mathrm{o}(1)$ steps, the same distortion bound is obtained on the larger sets $\{\Sigma^{(\le n)}\}$, but only with respect to average distortion. In Theorem 6 we show that the former class of functions (linear time automata) obtains the distortion bound on $\{\Sigma^{(\le n)}\}_{n=1}^{\infty}$ with respect to worst-case distortion.

We prepare the ground for the formal statement of Theorem 6. Throughout this section we consider automata whose output domain is $\Sigma^{(<\infty)}$. For such an automaton $\mathscr{A}$ and an input sequence a, we denote the output of $\mathscr{A}$ given a by $\mathscr{A}(a)$. We denote the number of states in $\mathscr{A}$ by $|\mathscr{A}|$. We use the notation δ_n, $\delta_{\le n}$, and D_n for $\delta_{\Sigma^{(n)}}$, $\delta_{\Sigma^{(\le n)}}$, and $D_{\Sigma^{(n)}}$, respectively.

Shannon's entropy is the following function:

$$H(\delta) = -\delta \log_2(\delta) - (1-\delta)\log_2(1-\delta),$$

for $0 \le \delta \le 1$.

The distortion-rate function $\delta(R)$ (in any one of the considered settings) will later be shown to be the smallest solution of the equation

$$H(\delta) = 1 - R,$$

for $0 \le R \le 1$. At the moment let us consider the above as the definition of $\delta(R)$.

The next proposition says that $\delta(R)$ is the (worst-case) distortion-rate function for $\{\Sigma^{(n)}\}_{n=1}^{\infty}$ and it is obtained by $\Sigma^{(n)}$ invariant functions.

Proposition 1. *For every* $0 < R \le 1$,

$$\delta(R) = \lim_{n\to\infty} \inf_{\substack{f:\Sigma^{(n)}\to\Sigma^{(<\infty)} \\ |f(\Sigma^{(n)})| \le 2^{Rn}}} \delta_n(f) \tag{1a}$$

$$= \lim_{n\to\infty} \min_{\substack{f:\Sigma^{(n)}\to\Sigma^{(n)} \\ |f(\Sigma^{(n)})| \le 2^{Rn}}} \delta_n(f). \tag{1b}$$

We can ask ourselves what the automaton complexity of the functions in Proposition 1 is. Neyman [4] has shown that these functions can be implemented through deterministic finite automata of the appropriate size that halt in linear time.

Theorem 5 (Neyman [4]). *For every* $0 < R \le 1$, *there exist deterministic finite automata* $\{\mathscr{A}_n\}_{n=1}^{\infty}$ *satisfying, for every n,*

1. $\mathscr{A}_n(\Sigma^{(n)}) \subset \Sigma^{(n)}$,
2. $|\mathscr{A}_n| \le 2^{Rn}$,
3. $\mathscr{A}_n$ *halts in n steps,*

and

$$\delta_n(\mathscr{A}_n) \xrightarrow[n\to\infty]{} \delta(R).$$

We extend the above theorem to the case where the exact cycle length is unknown.

THEOREM 6. *For every $0 < R \leq 1$, there exist deterministic finite automata $\{\mathscr{A}_n\}_{n=1}^{\infty}$ satisfying, for every $m \leq n$,*

1. $\mathscr{A}_n(\Sigma^{(\leq m)}) \subset \Sigma^{(\leq m)}$,
2. $|\mathscr{A}_n| \leq 2^{Rn}$,
3. $\mathscr{A}_n$ *halts in* $4n + 2\sqrt{n \log n}$ *steps,*

and

$$\delta_{\leq n}(\mathscr{A}_n) \xrightarrow[n\to\infty]{} \delta(R).$$

4.3. Proofs. Proposition 1 estimates the minimal number of balls needed to cover the n-dimensional Hamming space, $\Sigma^{(n)}$. The balls of radius $\delta(R)$ centered at the points of $f(\Sigma^{(n)})$ have to cover $\Sigma^{(n)}$. The question is how many balls of radius δ are needed to cover $\Sigma^{(n)}$, and the answer is $2^{(1-H(\delta))n+o(n)}$.

This kind of problem often appears in the context of information theory. The only difference from the standard theory is the fact that the center of the balls need not lie in $\Sigma^{(n)}$ but rather lies in a larger space, $\Sigma^{(<\infty)}$.

We would like to reduce Proposition 1 to statements about balls centered in $\Sigma^{(n)}$. To do so we consider the average distortion, $D_n(f)$, which is, by definition, a lower bound of $\delta_n(f)$. Our plan is to prove the following chain of inequalities:

$$\begin{aligned}
\delta(R) &\overset{\text{(i)}}{\leq} \liminf_{n\to\infty} \min_{\substack{f:\Sigma^{(n)}\to\Sigma^{(n)}\\ |f(\Sigma^{(n)})|\leq 2^{Rn}}} D_n(f)\\
&\overset{\text{(ii)}}{\leq} \liminf_{n\to\infty} \inf_{\substack{f:\Sigma^{(n)}\to\Sigma^{(<\infty)}\\ |f(\Sigma^{(n)})|\leq 2^{Rn}}} D_n(f)\\
&\overset{\text{(iii)}}{\leq} \liminf_{n\to\infty} \inf_{\substack{f:\Sigma^{(n)}\to\Sigma^{(<\infty)}\\ |f(\Sigma^{(n)})|\leq 2^{Rn}}} \delta_n(f)\\
&\overset{\text{(iv)}}{\leq} \limsup_{n\to\infty} \min_{\substack{f:\Sigma^{(n)}\to\Sigma^{(n)}\\ |f(\Sigma^{(n)})|\leq 2^{Rn}}} \delta_n(f) \overset{\text{(v)}}{\leq} \delta(R).
\end{aligned}$$

Inequalities (iii) and (iv) are obvious. Inequality (ii) stems from the fact that for every $f:\Sigma^{(n)} \to \Sigma^{(<\infty)}$,

$$D_n(f) = \lim_{T\to\infty} \operatorname*{average}_{\substack{a\in\Sigma^{(n)}\\ t\in\{1,\dots,T\}}} \mathbf{1}_{\{f(a)_t\neq a_t\}} = \lim_{T\to\infty} \operatorname*{average}_{t\in\{1,\dots,T\}} \operatorname*{average}_{\substack{a\in\Sigma^{(n)}\\ i\in\{1,\dots,n\}}} \mathbf{1}_{\{f(a)_{t+i}\neq a_{t+i}\}},$$

and for every $t \in \mathbb{N}$ there exists a (unique) function $f_t:\Sigma^{(n)} \to \Sigma^{(n)}$ that agrees with f in the coordinates $t+1, \dots, t+n$; thus,

$$\operatorname*{average}_{\substack{a\in\Sigma^{(n)}\\ i\in\{1,\dots,n\}}} \mathbf{1}_{\{f(a)_{t+i}\neq a_{t+i}\}} = D_n(f_t) \geq \min_{\substack{f':\Sigma^{(n)}\to\Sigma^{(n)}\\ |f'(\Sigma^{(n)})|\leq 2^{Rn}}} D_n(f').$$

The inequalities (i) and (v) refer to functions from $\Sigma^{(n)}$ to $\Sigma^{(n)}$ or equivalently to ball coverings of $\Sigma^{(n)}$. Let R: $[0, 1/2] \to [0, 1]$ be the inverse function of δ. That is,

$$R(\delta) = 1 - H(\delta).$$

By continuity, (i) and (v) are equivalent to the inequalities

$$\begin{aligned}
&\liminf_{n\to\infty} \min\left\{\frac{\log_2|F|}{n}: F \subset \Sigma^{(n)},\ \operatorname*{average}_{a\in\Sigma^{(n)}} d(a,F) \leq \delta\right\} \overset{\text{(i}')}{\geq} R(\delta)\\
&\overset{\text{(v}')}{\geq} \limsup_{n\to\infty} \min\left\{\frac{\log_2|F|}{n}: F \subset \Sigma^{(n)},\ \max_{a\in\Sigma^{(n)}} d(a,F) \leq \delta\right\},
\end{aligned}$$

for every $0 \leq \delta \leq 1/2$.

For the proof of (i′) and (v′) we will need an estimate of the size of the Hamming ball of radius δ in $\Sigma^{(n)}$, $B_n(\delta) = \sum_{0\leq k\leq \delta n} \binom{n}{k}$. The following asymptotic estimation will suffice (see Cover and Thomas [1, p. 353]):

$$\lim_{n\to\infty} \frac{1}{n} \log_2 B_n(\delta) = H(\delta), \tag{2}$$

for every $0 \leq \delta \leq 1/2$.

In the subsequent proofs of (i′) and (v′) we assume without loss of generality that $0 < \delta < 1/2$. The cases $\delta = 0$ and $\delta = 1/2$ hold trivially.

Proof of (i′). Let

$$F_n(\delta) \in \arg\min\left\{\frac{\log_2|F|}{n}\colon F \subset \Sigma^{(n)},\ \operatorname*{average}_{a\in\Sigma^{(n)}} d(a,F) \le \delta\right\}.$$

Let $0 < \epsilon < 1/2 - \delta$. Choose $a \in \Sigma^{(n)}$ uniformly at random. On the one hand, by Markov's inequality,

$$\Pr(d(a, F_n(\delta)) > \delta + \epsilon) \le \frac{\delta}{\delta+\epsilon} \le 1 - \epsilon.$$

On the other hand,

$$\Pr(d(a, F_n(\delta)) \le \delta + \epsilon) \le \frac{|F_n(\delta)|B_n(\delta+\epsilon)}{|\Sigma^{(n)}|}.$$

Combining the above inequalities gives

$$|F_n(\delta)| \ge \frac{\epsilon|\Sigma^{(n)}|}{B_n(\delta+\epsilon)}.$$

By taking $\log_2$ of both sides, dividing it by n, letting n grow infinitely and applying (2), we obtain

$$\liminf_{n\to\infty} \frac{\log_2|F_n(\delta)|}{n} \ge R(\delta+\epsilon).$$

The proof is concluded with the observation that the above holds for every $\epsilon > 0$ sufficiently small and the function R is continuous. □

Proof of (v′). Let $m_n = |\Sigma^{(n)}| \ln |\Sigma^{(n)}|/B_n(\delta)$. By (2), it is sufficient to prove the existence of sets $F_n(\delta) \subset \Sigma^{(n)}$ of size $|F_n(\delta)| \le m_n + 1$ such that $d(a, F_n(\delta)) \le \delta$, for every $a \in \Sigma^{(n)}$.

Let $x_1, \ldots, x_{\lceil m_n\rceil}$ be independent random variables that take values uniformly in $\Sigma^{(n)}$, and let $F_n(\delta) = \{x_1, \ldots, x_{\lceil m_n\rceil}\}$. It suffices to prove that

$$\Pr(\exists a \in \Sigma^{(n)} d(a, F_n(\delta)) > \delta) < 1.$$

Using the fact that $(1 - m^{-1})^m < e^{-1}$ for every $m \ge 1$, we have

$$\Pr(d(a, F_n(\delta) > \delta)) \le \left(1 - \frac{B_n(\delta)}{|\Sigma^{(n)}|}\right)^{m_n} < \frac{1}{|\Sigma^{(n)}|},$$

for every $a \in \Sigma^{(n)}$. Summing over every $a \in \Sigma^{(n)}$ concludes the proofs of (v′) and Proposition 1. □

The first step in the proof of Theorem 6 is a simple generalization of Neyman's theorem. Let $X_{k,n} = \bigcup_{1\le l\le n/k} \Sigma^{(lk)}$.

Lemma 1. *For every* $0 < R \le 1$, *there exist deterministic finite automata* $\{\mathcal{A}_{k,n}\}_{k,n=1}^{\infty}$ *satisfying, for every* $k \le n$,

1. $\mathcal{A}_{k,n}(\Sigma^{(lk)}) \subset \Sigma^{(lk)}$, *for every* $1 \le l \le [n/k]$,
2. $|\mathcal{A}_{k,n}| \le 2^{Rn}$,
3. $\mathcal{A}_{k,n}$ *halts in* $2n$ *steps,*

and

$$\delta_{X_{k,n}}(\mathcal{A}_{k,n}) \xrightarrow[k,n\to\infty]{} \delta(R).$$

Proof of Theorem 6 Assuming Theorem 1 and Lemma 1. Since the function δ is continuous, it is sufficient to construct automata $\{\mathcal{A}_n\}$ with $2^{Rn+o(n)}$ states.

We describe $\mathcal{A}_n$. Consider an input stream $a \in \Sigma^{(\le n)}$. Use the automaton provided by Theorem 1 to compute a number $k \ge \sqrt{n\log n}$ such that $\operatorname{lcm}(k, \rho(a)) \le n$; hence $a \in X_{k,n}$. Proceed with the automaton $\mathcal{A}_{k,n}$ provided by Lemma 1.

The running time is at most $2(n + \sqrt{n\log n}) + 2n$. The number of states needed is at most $2^{O(\sqrt{n\log n})} + n2^{Rn}$, counting the states that compute k plus the states of $\mathcal{A}_{k,n}$, for every possible value of k. □

The proof of Lemma 1 is a simple modification of Neyman's original proof [4, p. 24]. For completeness, we present a self-contained proof.

Proof of Lemma 1. For $k \le n$, we shall first describe the function induced by $\mathscr{A}_{k,n}$, and then construct the automaton itself and claim its properties.

Consider a sequence of functions $\{f_n \colon \Sigma^{(n)} \to \Sigma^{(n)}\}$ provided by Proposition (1b). Denote $\Sigma^{<\infty} = \bigcup_{n=0}^{\infty} \Sigma^n$ and define $\varphi \colon \Sigma^{<\infty} \to \Sigma^{<\infty}$ by

- $\varphi(\varnothing) = \varnothing$,
- $\varphi(x) = (f_n(x, x, \ldots)_t)_{t=1}^n$, for every $n \ge 1$ and $x \in \Sigma^n$.

Let $L = \min\{k, [\sqrt{n}]\}$ and $l = \lceil k/L \rceil$. Define $k_1, \ldots, k_l$ by $k_i = [k/l] + 1_{\{i \le k - l[k/l]\}}$. It should be noted that

- $k_1 + \cdots + k_l = k$, and
- $L/2 \le k_i \le L$, for every i.

For $t \in \mathbb{Z}_+$, define

$$r(t) = \min\{r \in \mathbb{Z}_+ \colon \exists\, 1 \le i \le l \quad \text{s.t. } t = k_1 + \cdots + k_i + r \bmod k\},$$

and

$$b(t) = \max\{b \le t \colon \exists\, 1 \le i \le l \quad \text{s.t. } b = k_1 + \cdots + k_i \bmod k\}.$$

Note that, for every $t \in \mathbb{Z}_+$,

- $t = b(t) + r(t)$,
- $r(t) \le L$,
- $b(t+1) - b(t) \in \{0, k_1, \ldots, k_l\}$,
- $b(t+1) \in \{b(t), t+1\}$.

We define an operator on infinite sequences:

$$A \colon \Sigma^\infty \to \Sigma^\infty,$$

$$A(x_1, x_2 \ldots) = (\varphi(x_{b(0)+1}, \ldots, x_{b(1)}), \ldots, \varphi(x_{b(t-1)+1}, \ldots, a_{b(t)}), \ldots).$$

In other words: $A(x)$ is the concatenation of the finite sequences $\varphi(x_{b(t-1)+1}, \ldots, a_{b(t)})$, $t = 1, 2, \ldots$.

Now we construct the automaton $\mathscr{A}_{k,n} = \langle \Sigma, S, \varnothing, f, H, X_{k,n}, g \rangle$, such that $\mathscr{A}_{k,n}(a) = A(a)$, for every $a \in X_{k,n}$. Since the function δ is continuous, it is sufficient to have $|\mathscr{A}_{k,n}| = 2^{Rn + o(n)}$.

The states are finite sequences of bits.

$$S = \{\varnothing\} \cup \{(A(a)_1, \ldots, A(a)_{b(t)}, a_{b(t)+1}, \ldots, a_{b(t)+r(t)}) \colon 1 \le t \le 2k[n/k], a \in X_{k,n}\}.$$

The initial state is the empty sequence,

$$s_* = \varnothing.$$

The terminal states are the longest sequences in S,

$$H = S \cap \Sigma^{2k[n/k]} = \{(A(a)_1, \ldots, A(a)_{2k[n/k]}) \colon a \in X_{k,n}\}.$$

The transition function is defined on nonterminal states and input streams in $X_{k,n}$,

$$f((x_1, \ldots, x_t), a) = \begin{cases} (x_1, \ldots, x_t, a), & \text{if } r(t+1) > 0; \\ (x_1, \ldots, x_{b(t)}, \varphi(x_{b(t)+1}, \ldots, x_t, a)), & \text{if } r(t+1) = 0. \end{cases}$$

Note that, for every $a \in X_{k,n}$ and $1 \le t \le 2k[n/k]$, the state of the automaton at time t given a, $s_t(a)$, is given by the expression in the definition of S,

$$s_t(a) = (A(a)_1, \ldots, A(a)_{b(t)}, a_{b(t)+1}, \ldots, a_{b(t)+r(t)}).$$

For completeness, we arbitrarily define the transition in the case that the above expression does not yield an element of S (it may happen only if either $t \ge k[n/k]$ or $a \notin X_{k,n}$).

For every $a \in X_{k,n}$, the run of the automaton on a halts in the state $(A(a)_1, \ldots, A(a)_{2k[n/k]})$. The operator A commutes with the k-places shift operator. Namely, $A(x_1, x_2 \ldots)_{t+k} = A(x_{k+1}, x_{k+2}, \ldots)_t$, for every $x \in \Sigma^\infty$ and $t \in \mathbb{N}$. Therefore, $A(\Sigma^{(km)}) \subset \Sigma^{(km)}$, for every $m \ge 1$. By Claim 2, the following equation well defines the output function on $X_{k,n}$:

$$g(A(a)_1, \ldots, A(a)_{2k[n/k]}) = A(a).$$

For every input stream a,

$$d(a, A(a)) \le \max_{\substack{1 \le i \le l \\ x \in \Sigma^{k_i}}} d(x, \varphi(x)) \le \sup_{m \ge L/2} \delta_m(f_m) \xrightarrow[k, n \to \infty]{} \delta(R).$$

Since $\mathscr{A}_{k,n}(a) = A(a)$, for every $a \in X_{k,n}$, we have $\limsup_{k,n} \delta_{X_{k,n}}(\mathscr{A}_{k,n}) \le \delta(R)$.

It remains to verify that $|S| = 2^{Rn+o(n)}$. The mapping $(x_1, \dots, x_t) \mapsto ((x_1, \dots, x_{b(t)}), (x_{b(t)+1}, \dots, x_t))$ maps S into a product of two sets, $S_1 \times S_2$, where

$$S_1 = \{(A(a)_s)_{s=1}^{t} : 0 \le t \le 2k[n/k], a \in X_{k,n}\},$$

$$S_2 = \bigcup_{t=0}^{L} \Sigma^t.$$

The cardinality of S_2 is $\sum_{t=0}^{L} |\Sigma|^t \le 2|\Sigma|^{\sqrt{n}}$. Consider the alphabet $\Gamma = [\varphi(\Sigma^{k_1}) \times \cdots \times \varphi(\Sigma^{k_l})]$. The definition of φ ensures that $|\Gamma| \le 2^{Rk}$. The set $A(X_{k,n})$ maps into $\Gamma^{(\le [n/k])}$; therefore $|S_2| \le (2n+1)|\Gamma^{(\le [n/k])}| \le 2^{Rn+\log_2 n+3}$. □

5. Game theoretic background. In this section we discuss the relevance of our results to the study of repeated games with finite automata.

Neyman [4] studies repeated two-person zero-sum games where each player is restricted to strategies that can be implemented through finite automata whose size is commonly known. In particular, he focuses on the case where one of the players is oblivious. An oblivious automaton with n states is equivalent to a periodic sequence whose cycle length is at most n. Neyman shows that if player 1 is oblivious and the game is repeated often enough, the asymptotic value of the game is given by a function $v(\log n_2/n_1)$, where n_i is the number of states in player i's automata.

Neyman constructs an automaton for player 2. In the first stage the automaton probabilistically learns the *exact* cycle length. To succeed with probability $1-\epsilon$, Neyman's automaton requires $C(\epsilon)n \log n$ steps of computation. In the second stage the automaton uses the cycle length to devise[5] a number $1 \ll k \le n$, such that k is a multiple of the cycle length; i.e., the input is k-periodic. Using the multiple of the cycle length, the automaton deterministically computes a "best reply" sequence in $O(n)$ (actually k) steps.

Theorem 6 improves Neyman's result (in the special case of the matching pennies game) by showing that the asymptotic value can be obtained using a deterministic automaton (pure strategy), guaranteeing that the play enters a cycle within $O(n)$ steps. The first stage in Neyman's construction is replaced by the automaton of Theorem 1. The second stage is modified so as to replace the requirement that k is divisible by the cycle length by the weaker requirement that the least common multiple of k and the cycle length is at most n. The requirement that $k \gg 1$ remains.

Neyman's function v is a generalization of the distortion-rate function, δ, described in §4. Proposition 1 says that $\delta(R)$ is the asymptotic value of a repeated matching pennies game in which player 1 chooses a sequence $a \in \Sigma^{(n)}$ and player 2 chooses a function $f\colon \Sigma^{(n)} \to \Sigma^{(n)}$ whose image contains at most 2^{Rn} points. For a general finite two-person zero-sum game, the asymptotic value is given by $v(R)$. By appropriately replacing $\delta(R)$ with $v(R)$ in §4, one can essentially recover and extend the results of [4], showing that a deterministic automaton of 2^{Rn} states can guarantee a payoff of at most $v(R) + o(1)$, and at the same time make sure that the play enters a loop in the first $(4 + o(1))n$ steps against any $(\le n)$-periodic sequence of actions of player 1.

Acknowledgments. The author thanks Professor Benjamin Weiss for his significant contribution to the proof of Theorem 1; see Footnote 1. The author would also like to thank Professor Michael O. Rabin for a fruitful conversation, Professor Eilon Solan for his comments, Professor Ehud Lehrer, and last, but not least, Professor Abraham Neyman for many helpful comments. This work is based on the author's Ph.D. Thesis done at the Center for the Study of Rationality, Hebrew University. It was supported in part by the Israel Science Foundation [Grant #212/09] and by the Google Inter-university center for Electronic Markets and Auctions and by the European Research Council under the European Community's Seventh Framework Programme (FPT/2007-2013)/[ERC Grant Agreement #249159].

References

[1] Cover TM, Thomas JA (2006) *Elements of Information Theory*, 2nd ed. (Wiley Interscience, New York).

[2] Karp RM, Rabin MO (1987) Efficient randomized pattern-matching algorithms. *IBM J. Res. Development* 31(2):249–260.

[3] Kushilevitz E, Nisan N (1997) *Communication Complexity* (Cambridge University Press, New York).

[4] Neyman A (2008) Learning effectiveness and memory size. Discussion Paper 476, Center for the Study of Rationality, Hebrew University, Jerusalem. http://ideas.repec.org/p/huj/dispap/dp476.html.

[5] In Neyman's construction, k is at least $n/2$, but any lower bound that diverges to infinity as n grows would do. This is what "$1 \ll k$" means.