Algorithm for the complexity of finite automata

Chih-Hung Chang
Department of Applied Mathematics
Feng Chia University, Taichung 40724, Taiwan, R.O.C.
chihhung@mail.fcu.edu.tw

Abstract

Testing the ambiguity of finite-state automata, in the viewpoint of symbolic dynamics, is related to the study of topological entropy. This elucidation proposes an algorithm for the computation of the topological entropy of a given finite-state automaton. Our algorithm is efficient for transforming a finite-state automaton into a deterministic finite-state automaton, and thus can be used to determine the topological entropy of the original finite-state automaton.

1 Introduction

A finite-state automaton $\mathcal{F}$ is a 5-tuple (Σ, Q, E, I, F) where Σ is a finite alphabet; Q is a finite set of states; $I \subseteq Q$ the set of initial states, and $E \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times Q$ a finite set of transitions, herein ε denotes the empty string. Mathematically, a finite-state automaton is a labeled graph with a distinguished "initial state" and a distinguished subset of "terminal states". A *language* is a set of words over a finite alphabet. The language of a finite-state automaton is the set of all labels of paths that begin at the initial state and end at a terminal state, and a language is called a *regular language* if it is a language of a finite-state automaton.

A finite-state automaton whose labeling is right-resolving is called a deterministic finite-state automaton or DFA for short. To investigate a finite-state automaton is to study the shift space induced by a labeled graph in symbolic dynamical systems. The connections between, regular languages, automata theory, and symbolic dynamics are referred to [4, 7, 9, 10].

$\mathcal{F}$ is said to be *unambiguous* if no string $x \in \Sigma^*$ labels two distinct accepting paths. Determining the ambiguity of $\mathcal{F}$ is related to the topological entropy of $\mathcal{F}$ and has been widely elucidated for the past few decades [1, 5, 6, 8, 12, 13, 14]. This paper aims to provide an algorithm to demonstrate the complexity of a finite-state automaton through calculating topological entropy.

The rest of the paper is organized as follows. Section 2 presents the definitions and some results of symbolic dynamics that are related to the study of finite-state automata. Our main results are revealed in Section 3, and Section 4 concludes the present paper with further work.

2 Preliminary

This section recall some definitions and known results for labeled graphs and sofic shifts to make the present elucidation self-contained. We refer the reader to [3, 2, 11] and the references therein for more details.

Let $\mathcal{A} = \{0, 1, \ldots, n-1\}$ be a finite alphabet with cardinality $|\mathcal{A}| = n$. The full $\mathcal{A}$-shift $\mathcal{A}^{\mathbb{Z}}$ is the collection of all bi-infinite sequences of symbols from $\mathcal{A}$. More precisely,

$$\mathcal{A}^{\mathbb{Z}} = \{\alpha = (\alpha_i)_{i \in \mathbb{Z}} : \alpha_i \in \mathcal{A} \text{ for all } i \in \mathbb{Z}\}.$$

The shift map σ on the full shift $\mathcal{A}^{\mathbb{Z}}$ is defined by

$$\sigma(\alpha)_i = \alpha_{i+1} \quad \text{for} \quad i \in \mathbb{Z}.$$

A *shift space* X is a subset of $\mathcal{A}^{\mathbb{Z}}$ such that $\sigma(X) \subseteq X$.

Definition 2.1. For each $m \in \mathbb{N}$, let

$$\mathcal{A}_m = \{w_0 w_1 \cdots w_{k-1} : w_i \in \mathcal{A}, 1 \leq k \leq m\}$$

and let $\mathcal{A}_0$ denote the empty set. If X is a shift space and there exists $L \geq 0$ and $\mathcal{F} \subseteq \mathcal{A}_L$ such that

$$X = \{(\alpha_i)_{i \in \mathbb{Z}} : \alpha_i \alpha_{i+1} \cdots \alpha_{i+k-1} \notin \mathcal{F} \\ \text{for } k \leq N, i \in \mathbb{Z}\},$$

then we say that X is a *shift of finite type* (SFT).

A SFT can be constructed via a finite, directed graph by considering the collection of all bi-infinite walks on the graph. We recall some definitions first.

Definition 2.2. A (directed) *graph* $G = (\mathcal{V}, \mathcal{E})$ consists of a finite set $\mathcal{V} = \mathcal{V}(G)$ of *vertices* (or *states*) together with a finite set $\mathcal{E} = \mathcal{E}(G)$ of *edges*. Each edge $e \in \mathcal{E}$ starts at an initial state $i(e)$ and terminates at a terminal state $t(e)$. Sometimes we also denote an edge e by $e = (i(e), t(e))$ to emphasize the initial and terminal states of e.

Let G and H be graphs. A *homomorphism* $(\partial\Phi, \Phi) : G \to H$ consists of a pair of maps $\partial\Phi : \mathcal{V}(G) \to \mathcal{V}(H)$ and $\Phi : \mathcal{E}(G) \to \mathcal{E}(H)$ such that $i(\Phi(e)) = \partial\Phi(i(e))$ and $t(\Phi(e)) = \partial\Phi(t(e))$ for all $e \in \mathcal{E}(G)$. A homomorphism is an *isomorphism* if both $\partial\Phi$ and Φ are one-to-one and onto.

Without loss of generality, we assume that, for any two vertices in a graph, there is at most one corresponding (directed) edge. Let $G = (\mathcal{V}, \mathcal{E})$ be a graph. The *transition matrix* T_G of G, indexed by $\mathcal{V}$, is an incidence matrix defined by $T_G(I, J) = 1$ if and only if $(I, J) \in \mathcal{E}$. On the other hand, suppose T is an $n \times n$ incidence matrix, then the graph of T is the graph $G = G_T$ with vertex set $\mathcal{V} = \{1, 2, \dots, n\}$, and with $T(I, J)$ edge from vertex I to vertex J. It follows immediately from the definitions that

$$T = T_{G_T} \quad \text{and} \quad G \cong G_{T_G}$$

Each graph G with corresponding transition matrix T gives rise to a SFT.

Definition 2.3. Let $G = (\mathcal{V}, \mathcal{E})$ be a graph with transition matrix T. The edge shift X_G or X_T is the shift space over the alphabet $\mathcal{A} = \mathcal{E}$ specified by

$$\begin{aligned} X_G = X_T = \{\xi = (\xi_j)_{j\in\mathbb{Z}} \in \mathcal{E}^{\mathbb{Z}} : \xi_j, \xi_{j+1} \in \mathcal{E} \\ \text{such that } i(\xi_{j+1}) = t(\xi_j) \text{ for } j \in \mathbb{Z}\}. \end{aligned} \tag{1}$$

Suppose G is a graph. It is seen that X_G is a shift of finite type. Sometimes certain edges of G can never appear in X_G, and such edges are inessential for the edge shift. A vertex $I \in \mathcal{V}$ is called *stranded* if either no edges start at I or no edges terminate at I. We say that a graph is *essential* if no vertex of the graph is stranded. The following proposition demonstrates that we can focus the discussion on those essential graphs.

Proposition 2.4 ([11])**.** *If G is a graph, then there is a unique subgraph H of G such that H is essential and $X_H = X_G$.*

A graph G is *irreducible* if for every ordered pair of vertices I and J there is a path in G starting at I and terminating at J, herein a path $\pi = v_1 v_2 \dots v_m$ on a graph G we mean a finite sequence of vertices from G such that $(v_i, v_{i+1}) \in \mathcal{E}$ for $i = 1, 2, \dots, m-1$. It can be verified that an essential graph is irreducible if and only if its edge shift is irreducible.

Suppose we label the edges in a graph with symbols from an alphabet $\mathcal{S}$. Every bi-infinite walk on the graph yields a point in the full shift $\mathcal{S}^{\mathbb{Z}}$ by reading the labels of its edges, and the set of all such points is called a *sofic shift*.

Definition 2.5. Suppose $G = (\mathcal{V}, \mathcal{E})$ is a directed graph, and $\mathcal{S}$ is a finite alphabet. A labeled graph $\mathcal{G}$ is a pair $(G, \mathcal{L})$ with graph G and the labeling $\mathcal{L} : \mathcal{E} \to \mathcal{S}$ assigns to each edge e of G a label $\mathcal{L}(e) \in \mathcal{S}$. The underlying graph of $\mathcal{G}$ is G.

Let $\mathcal{G} = (G, \mathcal{L}_G)$ and $\mathcal{H} = (H, \mathcal{L}_H)$ be labeled graphs. A *labeled-graph homomorphism* from $\mathcal{G}$ to $\mathcal{H}$ is a a graph homomorphism $(\partial\Phi, \Phi) : G \to H$ such that $\mathcal{L}_H(\Phi(e)) = \mathcal{L}_G(e)$ for all $e \in \mathcal{E}(G)$. A labeled-graph homomorphism is actually a *labeled-graph isomorphism* if $(\partial\Phi, \Phi)$ is an isomorphism.

Definition 2.6. Suppose $\mathcal{G} = (G, \mathcal{L})$ is a labeled graph. The shift space $X_{\mathcal{G}}$ is called a *sofic shift*. Moreover, we say that $\mathcal{G}$ is *right-resolving* if $\mathcal{L}((v, w)) \neq \mathcal{L}((v, w'))$ for $v \in \mathcal{V}$ and $(v, w), (v, w') \in \mathcal{E}$.

It is seen that a SFT is also a sofic shift. Indeed, sofic shifts is an extension of SFTs.

Theorem 2.7 ([11])**.** *A shift space is sofic if and only if it is a factor of a SFT. Furthermore, a sofic shift is a SFT if and only if it has a presentation $(G, \mathcal{L})$ such that $\mathcal{L}_\infty$ is a conjugacy.*

A quantity that describes the complexity of a system is *topological entropy*. Suppose X is a shift space. Denote $\Gamma_k(X)$ the cardinality of the collection of words of length k. The topological entropy of X is then defined by

$$h(X) = \lim_{k\to\infty} \frac{\Gamma_k(X)}{k}$$

3 Complexity of Finite Automata

One of the most frequently used quantum for the index of spatial complexity is the *topological entropy*. Notably, the topological entropy measures the growth rate of a number of patterns of an invariant closed space. Since a finite-state automaton is presented by a labeled graph $\mathcal{G} = (G, \mathcal{L})$ with a distinguished "initial state" and a distinguished subset of "terminal states", it is not an invariant space. By complexity, therefore, instead of the classical topological entropy, we focus on the following definition.

Definition 3.1. Suppose $\mathcal{F}$ is a finite-state automaton. Let $N_n(\mathcal{F})$ denote the number of distinct initial blocks of length n in $\mathcal{F}$. The topological entropy[1] of $\mathcal{F}$ is defined by

$$h_f(\mathcal{F}) = \limsup_{n\to\infty} \frac{1}{n} \log N_n(\mathcal{F}). \tag{2}$$

Our main result then follows.

Theorem 3.2. *For a finite-state automaton $\mathcal{F}$,*

$$h_f(\mathcal{F}) = h_{top}(X_{\mathcal{G}}) \tag{3}$$

provided $\mathcal{G}$ is irreducible, where $\mathcal{G}$ is a labeled graph representation of $\mathcal{F}$.

Theorem 3.2 indicates that, once the labeled graph representation $\mathcal{G}$ of a finite-state automaton $\mathcal{F}$ is given and is irreducible, the computation of topological entropy $h_f(\mathcal{F})$ is equivalent to find the classical topological entropy of a sofic shift $X_{\mathcal{G}}$. Let X be a SFT with transition matrix T. Perron-Frobenius Theorem indicates that $h(X) = \log \rho(T)$, where $\rho(T)$ is the spectral radius of T. Nevertheless, if X is a sofic shift which is not right-resolving, then $\log \rho(T)$ might no longer be the topological entropy of X. Instead, we need to find X a right-resolving presentation via the so-called *subset construction method* (SMC).

Subset Construction Method

Let X be a sofic shift over the alphabet $\mathcal{A}$ having a presentation $\mathcal{G} = (G, \mathcal{L})$. If $\mathcal{G}$ is not right-resolving, then a new labeled graph $\mathcal{H} = (H, \mathcal{L}_H)$ is constructed as follows.

The vertices I of H are the nonempty subsets of the vertex set $\mathcal{V}(G)$ of G. If $I \in \mathcal{V}(H)$ and $a \in \mathcal{A}$, let J denote the set of terminal vertices of edges in G starting at some vertices in I and labeled a, i.e., J is the set of vertices reachable from I using the edges labeled a.

1) If $J = \varnothing$, do nothing.

2) If $J \neq \varnothing$, $J \in \mathcal{V}(H)$ and draw an edge in H from I to J labeled a.

Carrying this out for each $I \in \mathcal{V}(H)$ and each $a \in \mathcal{A}$ produces the labeled graph $\mathcal{H}$. Then, each vertex I in H has at most one edge with a given label starting at I. This implies that $\mathcal{H}$ is right-resolving.

Theorem 3.3 ([11])**.** *Let $\mathcal{G} = (G, \mathcal{L})$ be a labeled graph representation of a sofic shift X which is not right-resolving, and let $\mathcal{H} = (H, \mathcal{L}_H)$ be a right-resolving labeled graph constructed under the subset construction method. Then $X_{\mathcal{G}} = X_{\mathcal{H}}$.*

It follows immediately that $h_{top}(X) = h_{top}(X_{\mathcal{H}}) = \log \rho(T_{\mathcal{H}})$. In other words,

$$h_f(\mathcal{F}) = h_{top}(X_{\mathcal{G}}) = \log \rho(T_{\mathcal{H}}) \tag{4}$$

provided $\mathcal{G}$ is a labeled graph representation of finite-state automaton $\mathcal{F}$ and $\mathcal{H}$ is a right-resolving labeled graph obtained by applying SMC to $\mathcal{G}$.

Suppose M is the *symbolic transition matrix* of a labeled graph $\mathcal{G}$, namely,

$$M(p,q) = \begin{cases} \mathcal{L}(e), & i(e) = p, t(e) = q,\ T(p,q) = 1; \\ \varnothing, & \text{otherwise.} \end{cases} \tag{5}$$

Herein $e \in \mathcal{E}$. Now we are ready to propose the algorithm for measuring the complexity of a finite-state automaton $\mathcal{F}$. Let $k = |\mathcal{S}|$ and $\ell = |\mathcal{V}|$ denote the cardinality of the set of labels $\mathcal{S}$ and vertices $\mathcal{V}$, respectively..

[1] We abuse the terminology "topological entropy" for the measurement of complexity of a finite-state automaton without ambiguity.

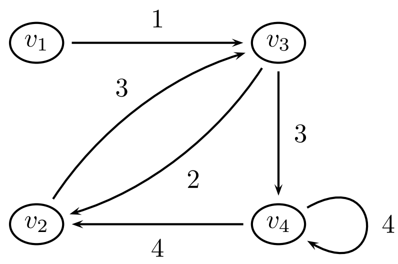


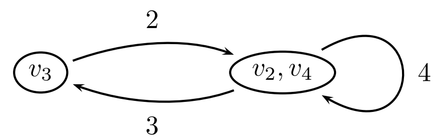
Figure 1: The graph presentation of the finite-state automaton in Example 3.4.

Algorithm EFA(M)

for i is between 1 and k **do**
  $L_i \leftarrow$ projection of M with symbol indexed i
  enlarge L_i to restore all nonempty subset of $\mathcal{S}$
  for j is between 1 and 2^ℓ **do**
    if $(U \subseteq \mathcal{V}$ and $L_i(U) = i)$ **then**
      $L_i(j, U) \leftarrow 0$ and $L_i(j, \{U\}) \leftarrow i$
      $V(i, \{U\}) \leftarrow i$
    end if
  end for
end for
$N \leftarrow \Sigma L_i$
Construct a zero matrix Vtest
for i is between 1 and ℓ **do**
  while $\Sigma_j V(i,j) > \Sigma_j Vtest(i,j)$ **do**
    $Vtest \leftarrow V$
    for j is between $\ell + 1$ and 2^ℓ **do**
      if $(V(i,j) > 0)$ **then**
        $V(i) \leftarrow V(i) + V(j)$
      end if
    end for
  end while
end for
$V(1) \leftarrow \Sigma_{1 \leq j \leq \ell} V(i)$
for i is between 1 and 2^ℓ **do**
  if $(V(1,i) = 0)$ **then**
    $N(i) = 0$
  end if
end for
$\rho \leftarrow$ maximal eigenvalue of N
entropy $\leftarrow \log \rho$

Example 3.4. Suppose a finite-state automaton $\mathcal{F}$ comes with labeled graph representation $\mathcal{G}$ with initial state v_1 and terminal states v_3, v_4. See Figure 1.

It is seen that $\mathcal{G}$ is not right-resolving. After applying SCM to $\mathcal{G}$, the irreducible right-resolving labeled graph representation $\mathcal{H}$ of $\mathcal{F}$ is shown as follows.



It comes immediately that the transition matrix of $\mathcal{H}$ is

$$T_{\mathcal{H}} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$$

with spectral radius $g = \dfrac{1+\sqrt{5}}{2}$, which is the golden mean. Theorem 3.2 infers that the topological entropy of $\mathcal{F}$ is $h_f(\mathcal{F}) = \log g$.

4 Conclusion

A finite-state automaton is, in the viewpoint of mathematics, a labeled graph with a distinguished "initial state" and a distinguished subset of "terminal states". This paper demonstrates that the topological entropy of a finite-state automaton is identical to the classical topological entropy of the sofic shift induced by its labeled graph representation. Furthermore, we propose an algorithm to find its corresponding deterministic finite-state automaton if the original one is not deterministic. The topological entropy of the finite-state automaton is also explicitly formulated.

An efficient algorithm for minimizing the number of the states in a specific finite-state automaton is under going.

Acknowledgment

This work is partially supported by the National Science Council, ROC (Contract No NSC 102-2115-M-035-004-).

References

[1] C. Allauzen, M. Mohri, A. Rastogi, and O. H. Ibarra. General algorithms for testing the ambiguity of finite automata and the double-tape ambiguity of finite-state transducers. *Theor. Comput. Sci.*, 22:883–904, 2011.

[2] J.-C. Ban and C.-H. Chang. Inhomogeneous lattice dynamical systems and the boundary effect. *Bound. Value Probl.*, 2013:249, 2013.

[3] J.-C. Ban and C.-H. Chang. The learning problem of multi-layer neural networks. *Neural Networks*, 46:116–123, 2013.

[4] M.-P. Béal. *Codage Symboliqu.* Masson, Paris, 1993.

[5] T.-H. Chan and O. H. Ibarra. On the finite-valuedness problem for sequential machines. *Theor. Comput. Sci.*, 23:95–101, 1983.

[6] C. Cortes, M. Mohri, A. Rastogi, and M. Riley. On the computation of the relative entropy of probabilistic automata. *Int. J. Founda. Comput. Sci.*, 19:219–241, 2008.

[7] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, 1979.

[8] G. Jacob. Un algorithme calculant le cardinal, fini ou infini, des demi-groupes de matrices. *Theor. Comput. Sci.*, 5:183–202, 1977.

[9] W. Krieger. On sofic systems I. *Israel J. Math.*, 48:305–330, 1984.

[10] W. Krieger. On sofic systems II. *Israel Journal of Mathematics*, 60:167–176, 1987.

[11] D. Lind and B. Marcus. *An introduction to symbolic dynamics and coding.* Cambridge University Press, Cambridge, 1995.

[12] A. Mandel and I. Simon. On finite semigroups of matrices. *Theor. Comput. Sci.*, 5:101–111, 1977.

[13] B. Ravikumar and O. H. Ibarra. Relating the type of ambiguity of finite automata to the succintness of their representation. *SIAM J. Comput.*, 18:1263–1282, 1989.

[14] A. Weber and H. Seidl. On the degree of ambiguity of finite automata. *Theor. Comput. Sci.*, 88:325–349, 1991.