The Complexity and Entropy of Turing Machines

Paul A. Dufort and Charles J. Lumsden
Department of Physics and Department of Medicine
University of Toronto
Toronto, Canada

Abstract

This paper addresses the relationship between dynamical systems theory and theoretical computer science, in particular the dynamical, information-theoretic and computational properties of systems that compute. These properties have been studied in cellular automata and the symbolic dynamics of maps over the unit interval [5-8, 14-15], but have never been addressed in compact systems known to be capable of universal computation. Recent work is described in which the entropy, periodicity and regular language complexity of a large number of randomly generated Turing machines were calculated. The results are discussed in detail and compared with an identical analysis of a universal Turing machine. This comparison yields the first direct quantitative evidence that universal computation lies between ordered and chaotic behavior. The discussion concludes with a list of questions remaining to be answered about the phase-space portrait of computationally complex systems.

1. Introduction

Dynamical systems theory has become an integral part of how we understand the time evolution of nature's physical systems. Fixed points and periodic attractors, transients, and basins of attraction are the root metaphors and the quantitative backbone in terms of which many analyses in physics, chemistry and biology are phrased. When a fundamentally different mode of dynamical systems behavior is discovered, the effect on the scientific community can be electrifying. Note, for example, the discovery and elucidation of chaos over the past several decades and the profound effect it has had on our understanding of many natural phenomena. Over the same period that chaos was being discovered, however, another science was being born which is also concerned with dynamical systems of a special kind: theoretical computer science.

Despite the ubiquity of dynamical systems theory and theoretical computer science, little is known about how these two theories relate to each other. In particular, what are the computational properties of a given dynamical system, and conversely, what does the phase-space portrait of a computer look like? Is it highly ordered, or highly chaotic, or does it belong to an entirely new class of dynamical behavior different from anything yet encountered? Recent work by Langton [14-15], Crutchfield [5-8] and others has suggested that dynamics capable of supporting complex computations occur at a phase transition separating ordered and chaotic behavior. However, there has been no direct quantitative evidence that, in the spectrum of computational systems, universal computation is found at this transition. The work described here addresses this point.

We begin with a review of the works by Langton and Crutchfield that have been crucial to the assertion that computation lies at a phase transition between order and chaos. Some central results from theoretical computer science are also reviewed as background to Crutchfield's work. We then describe our own work, which employs a combination of both Langton's and Crutchfield's techniques to analyze a large number of different Turing machines, only one of which is capable of universal computation. Finally, we conclude with a discussion of some outstanding questions.

2. Background

2.1 Langton's λ-parameter and the transition from order to chaos

Langton [14-15] has explored the relationship between dynamical systems theory and computation by studying cellular automata (CA) as a model system. In this work, a large number of 1- and 2-dimensional CA were randomly generated and simulated for at least 10^3 time steps, during which the dynamics were studied visually and the temporal and spatial entropy and mutual information were calculated and stored. In addition, the transient length was recorded, measuring the number of time steps elapsed before the dynamics either fell into a periodic orbit or converged to within 1% of their long time probability distribution. The CA were parametrized

0-8186-6715-X/94 $4.00

by the quantity λ, representing the number of input neighborhoods *not* mapping to a randomly chosen state called the "quiescent state". The λ parameter thus measures the level of bias in a given rule table towards the quiescent state.

The dynamics of the CA showed a strong correlation with the parameter λ. At low values of λ, the CA quickly reached a fixed point or entered a periodic orbit, and the entropy was minimal. At high values, the CA quickly entered a state of total disorder, perturbations propagated at the maximum rate possible, and the entropy was maximal, indicating a chaotic state. Approaching λ=0.5 from below, the transient length displayed a sharp peak, the entropy rose sharply, and the mutual information peaked and fell gradually. Visual inspection of the evolving CA revealed that the most complex behavior, defined by virtue of its being the most qualitatively unpredictable, occurred in the vicinity of λ=0.5. Langton conjectured that a system capable of universal computation could not be simply periodic or chaotic, and used his results to argue that the most computationally complex behavior of the CA must be occurring at the transition value of λ=0.5, between order and chaos. In support of this conjecture, the plot of mutual information versus entropy revealed low values of mutual information at low entropy (order) and at high entropy (chaos), while at middle values of entropy the mutual information was higher.

While extremely provocative, the force of Langton's conclusions are limited by one principle shortcoming. Although the qualitative arguments for computational complexity at the transition are persuasive, no quantitative evidence has been given to support this conclusion. Indeed, since it is precisely the dynamical signature of computation that this work was attempting to identify, it seems premature to conclude in advance that a particular kind of dynamic is computation without showing that it conforms to any known standard of computation, or showing that it can in fact compute anything. This limitation is exacerbated by the tenuous relationship between CA and computation: while it is known that certain CA are capable of universal computation (defined below), for example Conway's "Game of Life" [1], the size of the CA lattice required to actually build a functioning universal computer is astronomical. In the present work, we have attempted to remedy this problem by applying an analysis similar to Langton's on a much more compact system whose computational abilities are known and extensively documented, the Turing machine.

2.2 Theoretical computer science, Crutchfield's ε-machine reconstruction technique, and the logistic map

Theoretical computer science comprises the analysis of the information processing capabilities of abstract mathematical systems called formal grammars [11]. These systems consist in a set of symbols forming a finite alphabet, and a set of production rules which transform sequences of symbols from the alphabet into new sequences. The production rules are applied recursively to a string of symbols to transform it repeatedly until it reaches a final, irreducible form. Formal grammars are classified according to the complexity of their production rules into four classes, forming the Chomsky hierarchy. Each class subsumes all grammars in the classes below it in the hierarchy. In order of increasing complexity, the classes forming the Chomsky hierarchy are regular, context-free, context-sensitive, and unrestricted. Each of these classes has associated with it a device capable of recognizing strings generated from grammars in that class. These are: for regular grammars, the finite automaton (FA), represented by a labeled digraph; for context-free grammars, the pushdown automaton, represented by a labeled digraph augmented with a pushdown stack; for context-sensitive grammars, the linear bounded automaton, a Turing machine on a finite tape; and for unrestricted grammars, the Turing machine (TM) itself. Analysis of the properties of grammars becomes progressively more difficult as one ascends the Chomsky hierarchy. In fact, for Turing machines there is in general no finite procedure for determining if the Turing machine will even halt on a given input (the halting problem [11]). The pinnacle of the Chomsky hierarchy is occupied by the universal Turing machine (UTM), defined by its ability to simulate the behavior of any other Turing machine. Any finite algorithm that can be executed by a modern digital computer can be executed by a UTM - their computational abilities are identical.

Formal grammars represent the framework in terms of which the very notion of computation is rigorously defined, at least as the term is presently understood. Any dynamical system purported to be capable of computation must ultimately be tested against and classified within the Chomsky hierarchy if its computational abilities are to be rigorously and quantitatively defined. Fortunately, Crutchfield has developed a means for performing precsely this type of analysis, at least at the level of regular grammars.

Crutchfield [5-8] has addressed the issue of computation in dynamical systems by analyzing the computational abilities of the class of 1-dimensional

maps over the unit interval. In particular, the dynamics of the logistic map $f(x)=rx(1-x)$ have been analyzed by partitioning its trajectories into a sequence of the symbols 0 and 1, each representing half of the unit interval. The principal result of this analysis is a plot of regular language complexity C versus metric entropy H. The entropy is the Shannon information of all blocks of 16 symbols appearing at any point in the trajectory of the map. Since it has been shown that this quantity converges from above to the sum of the positive Lyapunov exponents of the system (under appropriate conditions) [4], this quantity is an indicator of the level of chaos in the trajectory: low values indicate fixed point or periodic behavior, while high values indicate chaotic behavior.

The language complexity is determined using Crutchfield's process of ε-machine reconstruction, a technique that parses a sequences of symbols from a dynamical system and deduces from it the simplest regular grammar capable of producing all symbol sequences generated by the system. The result is a labeled digraph representing the finite automaton corresponding to the minimal regular grammar. A set of normalized probabilities is defined over the nodes of the graph by counting the number of times the parsed trajectory passes through each of the nodes. The complexity of the graph is then calculated as the Shannon information of the node probabilities [17].

The complexity and entropy were calculated for many different values of the parameter r of the logistic map and plotted against each another [6,8]. At the minimum and maximum values of the entropy, corresponding to the periodic and fully chaotic regimes of the map, the language complexity approached zero. As intermediate values of the entropy were approached from below or from above, the complexity increased and displayed a discontinuous jump indicating the presence of a phase transition. The highest value of the complexity occurred for the parameter value of r where the period-doubling cascade of the map meets the band-merging cascade, precisely at the border between order and chaos. By examining the structure of the diverging regular grammars produced at that point, the dynamic of the map at the critical value of r was identified as corresponding to an indexed grammar [8], occupying a position between context-free and context-sensitive grammars in the Chomsky hierarchy.

This result provides solid evidence that maximal levels of computational complexity in 1-dimensional maps occur at a phase transition between order and chaos. However, it has only addressed this question up to the level of indexed grammars, whose computational abilities are still far below those of the universal Turing machine. The question of where in the spectrum between order and chaos universal computation is found has been left unanswered.

We have addresses this question by combining both Langton's and Crutchfield's methods of analysis and applying them to the class of 7-state 4-symbol Turing machines, whose members include Minsky's universal Turing machine [16], the smallest known UTM.

3. Methods

The 7-state 4-symbol Turing machine consists in a tape divided into cells, each of which contains one of the 4 symbols, and a read/write head capable of moving back and forth over the tape and writing new symbols. The read/write head is attached to a finite state device with 7 states. The machines are defined by three 4X7 transition tables, mapping the current state and symbol being read to: (i) a new symbol to be written; (ii) a new state to be entered; and (iii) a movement of the read/write head one cell to the left or right [11].

A large number ($5X10^4$) of these machines were randomly generated. Each machine was positioned at the centre of a tape 2000 cells long which had been randomly filled with symbols, and was simulated for 1000 time steps. The choice of random initial tapes is justified by the algorithmic information theory of Kolmogorov and Chaitin [2-3, 12-13]. This theory stipulates that any TM program which does not have high algorithmic complexity (i.e. is maximally compressed) can be converted to one that does have high algorithmic complexity. Since random strings are the most algorithmically complex TM programs possible, running a TM on a random string is a good representation of the TM executing a complex program.

During simulation of the TMs, each symbol written, state entered, and movement of the machine were recorded as three separate dynamical sequences. These three sequences together comprise the complete spatial and temporal history of a given Turing machine simulation, and can be used to reconstruct exactly the output of the machine (the input cannot be reconstructed in this way unless the transition tables are one-to-one). 2000 separate simulations were carried out for each machine, producing symbol, state, and movement dynamical sequences, each $2X10^6$ characters in length. Throughout each simulation the dynamics were tested for the existence of periodic orbits up to period 40, and the transient length prior to entering the orbits was recorded. These latter data were then compiled to determine the fraction of time spent by each machine in periodic orbits and in transients. The above procedure was also carried out for Minsky's universal machine.

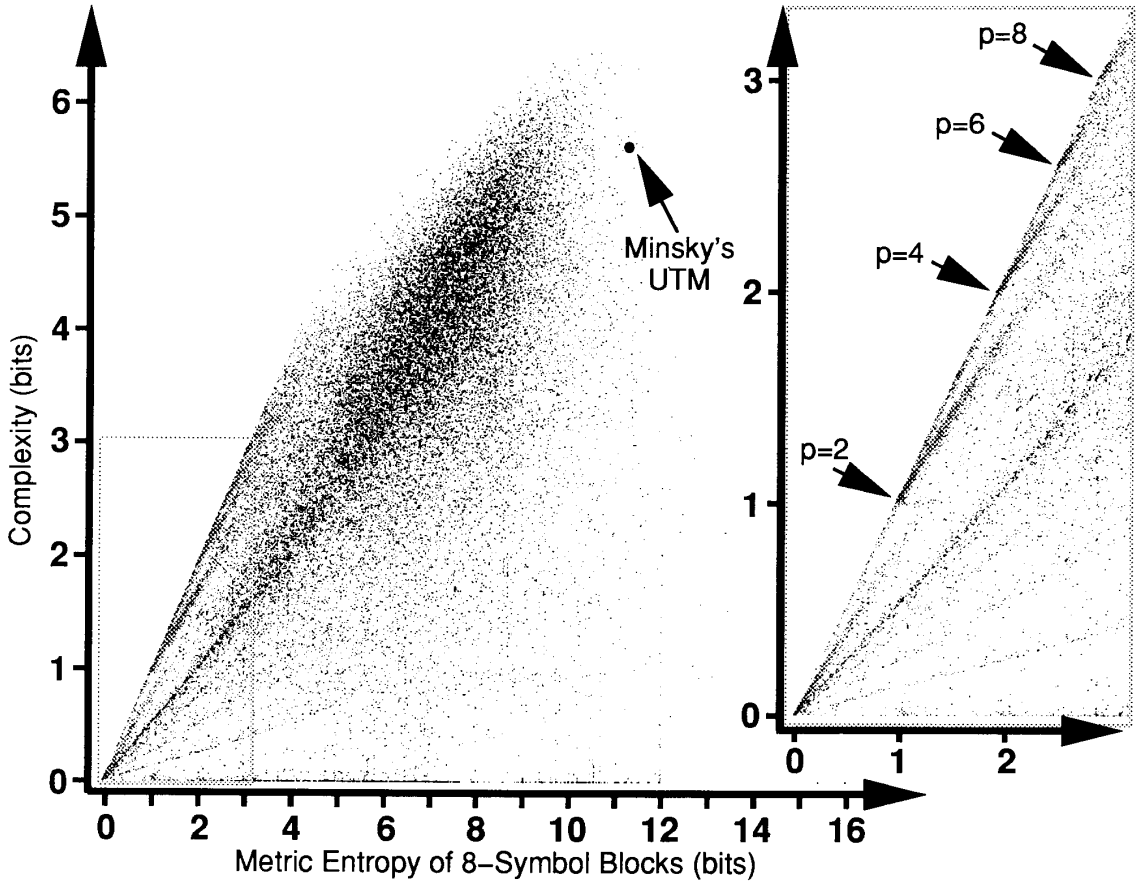

Figure: Regular language complexity versus metric entropy of 8-symbol blocks for the sequence of symbols written by Turing machines. Each dot represents one TM. The complexity is low at both low (ordered) and high (chaotic) values of the entropy. The highest values of complexity are found at intermediate values of the entropy. Minsky's universal Turing machine is also found in this region. Inset: Magnification of the box on the lower left of the figure. The marked points are the loci corresponding to TMs with periodic attractors of period 2, 4, 6 and 8. All purely periodic machines with no transients fall along a line of slope 1 at the extreme left edge of the graph. As one ascends this line the periods increase. TMs only admit attractors of even period, so this is where most of the periodic machines are clustered. Many machines possess several attractors with different periodicities, however, which appear between the loci of even periods. Emanating from each even-period locus at a slope just less than 1 is a stream of TMs with the periodicity of the source locus but with increasingly long transients prior to entering the attractor.

4. Results

The sequences of symbols written, states entered and movements made by each Turing machine were analyzed in two complementary ways [9]. The entropy of each sequence was determined by counting the number of occurrences of each possible block of length 8 for the symbol sequence, 6 for the state sequence, and 16 for the movement sequence (these limits were imposed by the amount of available computer memory). The regular language complexity of each sequence was determined using Crutchfield's ϵ-machine reconstruction technique.

(Although it could in priciple be problematic to model the behavior of TMs, at the top of the Chomsky hierarchy, with regular grammars, at the bottom of the Chomsky hierarchy, we have found in practice that this is

not a concern. Our experience has been that the behavior of randomly generated TMs ranges over the entire spectrum of regular language complexity, rather than all being clustered at the maximum complexity as one might naively expect. In addition, the graph indeterminacy of the ε-machine reconstructions, a measure of the extent to which the reconstructed regular grammar is unable to capture the details of the sequence [8], was consistently near zero. While a regular grammar clearly cannot model exactly the behavior of a TM, regular language complexity is nonetheless a excellent measure of each TM's computational power. The conclusion to be drawn from this result is that the majority of randomly generated TMs are not very complex, and are certainly not universal computers.)

The regular language complexity of the written symbol sequence is plotted against the entropy in the Figure. Each dot represents one TM. The inset in the upper-right is a magnification of the dotted box on the lower left. This distribution has several interesting features. All purely periodic machines with no transients fall along a line of slope 1 at the extreme left edge of the graph [6,8]. As one ascends this line the periods increase (inset). TMs only admit attractors of even period, so this is where most of the periodic machines are clustered. Many machines possess several attractors with different periodicities, however, which appear between the loci of even periods. Emanating from each even-period locus at a slope just less than 1 is a stream of TMs with the periodicity of the source locus but with increasingly long transients prior to entering the attractor (inset). The locus of periodic machines at the origin correspond to TMs whose attractor consisted in writing the same symbol repeatedly, but changing states at each point in the orbit.

At the lower left corner of the distribution several lines of fractional slope can be seen emanating from the origin. We have analyzed these lines [9] and have found that they belong to a class of TMs that simply erase the tape, filling it with one symbol interrupted only very infrequently with a second symbol. These sequence of written symbols from these machines is homologous to a 2-state Markov process with one state having probability close to 1, the other close to 0 [9]. In the lower right corner of the graph are the chaotic machines, few in number, that randomly produce all possible sequences of symbols with equal probability. Most of these machines are simply transcribing the random tape. The machines with the highest complexity occur at intermediate values of the entropy. The sequence of written symbols for these machines is neither completely ordered nor completely random, but is rather a complex combination of allowed and disallowed subsequences. The complexity-entropy plots for the state sequence and movement sequence showed similar features.

It is noteworthy that the mutual information between characters of the sequences and characters 1, 2, 4, or 8 sequences later was not a good indicator of computational sophistication: there was no correlation between regular language complexity and mutual information. Instead, mutual information tended to be highest for simple periodic machines.

The dynamical features of Minsky's universal TM were distinctive [10]. The symbol sequence for this machine was extremely complex and had intermediate values of entropy, but its precise position in the complexity-entropy plane is occupied by only a few randomly generated machines. More interestingly, the state sequence for the UTM was only moderately complex, and the movement sequence had a very low complexity. This is straightforward to observe visually if one watches the UTM in action. In contrast, some TMs have symbol, state, and movement complexities that are all maximal. The question remains as to whether these machines are in some way more computationally complex than the UTM (although Church's thesis stipulates that there is no TM with greater computational ability than the UTM [11]), or whether it is necessary for the UTM to keep two of its three operations (state entered and movement made) relatively simple in order to maintain sufficient coherence to compute effectively.

5. Discussion and conclusions

The work described here provides the first concrete evidence that the type of dynamics underlying the most sophisticated forms of computation, including universal computation, are neither simply ordered or fully chaotic but rather occur at intermediate values of entropy between order and chaos. However, this is a coarse characterization of the dynamics, and is only one of the first steps in the elucidation of the precise phase-space portrait of a computer. We believe that one of the greatest problems in the determination of this portrait arises from the fact that computation is usually defined by design, not by observation. Indeed, it is interesting to ask whether or not a physicist who came across an unfamiliar device performing a computation would be able to tell that a computation was being performed. He or she would likely be unable to tell exactly what was being computed, since this depends on the subjective assignment of meanings to the various values of the input variables. But could he or she identify that a computation was being performed at all?

There are several specific questions to be answered about the phase-space portrait of a computer. Clearly,

there must be a countably infinite number of fixed points (for Turing machines with halting states [11]) corresponding to the countably infinite number of possible outputs of a Turing machine. But how are these distributed throughout the phase-space, and what is the distribution of basin sizes and transient lengths leading up to the fixed points? If computation is not fully chaotic, then what is its sensitivity to initial conditions? Is it still exponential, or is it a power-law? And to what extent do computational trajectories leading to different results pass close to each other? What role does dissipation play? While it has been shown by Fredkin through his discovery of nondissipating logic gates that dissipation is not necessary for computation, to what extent is dissipation part of the behavior of complex Turing machines? And is rate of dissipation constant throughout the phase space, or it is heterogeneously distributed? (We are currently investigating the extent of dissipation in Turing machines and its correlation with their complexity and entropy [9]). What is the phase-space signature of specific but common algorithmic constructs such as recursion? These questions and many others must be answered before we will be able to observe nature's many complex dynamical systems and assess their computational properties.

References

[1] Berlekamp, E., Conway, J.H., and Guy, R. (1982) *Winning ways for your mathematical plays*. Academic Press, New York.

[2] Chaitin, G. (1966) On the length of programs for computing finite binary sequences. *J. ACM*. 13:145.

[3] Chaitin, G. (1975) Randomness and scientific truth. *Sci. Am.* May:47.

[4] Crutchfield, J.P. (1983) Symbolic dynamics of noisy chaos. *Physica 7D*. 201-223

[5] Crutchfield, J.P. (1989a) Inferring the dynamic, quantifying physical complexity. *Quantitative Measures of Complex Dynamical Systems*, ed. by N.B. Abraham. Plenum Press, New York.

[6] Crutchfield, J.P. andYoung, K. (1989b) Inferring statistical complexity. *Phys. Rev. Lett.* 63:105-108.

[7] Crutchfield, J.P. (1990a) Reconstructing language hierarchies. *Information Dynamics*, ed. by H.A. Atmanspracher. Plenum Press, New York.

[8] Crutchfield, J.P. andYoung, K. (1990b) Computation at the onset of chaos. *Complexity, Entropy, and the Physics of Computation*, ed. by W.H. Zurek. Addison-Wesley, New York.

[9] Dufort P.A. and Lumsden, C.J. (1994a) Computational complexity and dynamics of Turing machines. *In preparation.*

[10] Dufort P.A. and Lumsden, C.J. (1994b) Dynamical behavior of a universal Turing machine. *In preparation.*

[11] Hopcroft, J.E. and Ullman, J.D. (1979) *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, New York.

[12] Kolmogorov, A.N. (1965) Three approaches to the concept of the amount of information. *Prob. Info. Trans.* 1:1.

[13] Kolmogorov, A.N. (1983) Combinatorial foundations of information theory and the calculus of probabilities. *Russ. Math. Surveys*. 38:29.

[14] Langton, C.G. (1990) Computation at the edge of chaos. *Physica D*. 42:12-37.

[15] Langton, C.G. (1991) Computation at the edge of chaos: phase transitions and emergent computation. *Ph.D. Thesis*. UMI Dissertation Services.

[16] Minsky, M. (1967) *Computation: Finite and Infinite Machines*. Prentice-Hall, Englewood Cliffs, N.J.

[17] Shannon, C.E. and Weaver, W. (1949) *The Mathematical Theory of Communication*. University of Illinois Press, Urbana.