The Busy Beaver Game and the Meaning of Life

Allen H. Brady

1. Introduction

The representation of the Universal Computing Machine in the guise of the stored-program digital computer is now well known among serious students of computer science. On the periphery, a modern generation of technologists seems unable to conceive of a time when "computer" meant "a person who computes" and the concept of "programming" was not ubiquitous. For practical reasons, a modern computer is much more complex than is minimally necessary to achieve universality. The original formulation of a Universal Computing Machine (Turing *1936-7*) involved a table based upon 15 symbols and 28 states with no restriction imposed on the length of a sequence of atomic acts ("move" and "print") permitted prior to a change in state. An aboriginal use of *macros* simplified the description.

Briefly stated, the Universal Computing Machine is an active finite-state device of limited size connected to a passive medium of unlimited extent. The active device writes on and reads from the passive medium. Any other finite-state machine of any size or complexity whatsoever, even one connected to its own passive recording medium, may be reduced to nothing more than an abstraction recorded in the medium of the Universal Machine. The Universal Machine without any change in its mechanism then assumes the identity of the machine which has been described to it.

A competition to find the *smallest* Universal Turing Machine received at one time a certain amount of attention (cf. Minsky *1967*). The measure of "smallness" proposed was the size of the state-symbol product in the machine description. A clever construction by Shannon *1956* had demonstrated that with enough symbols any Turing machine can be reduced to only two states. This demonstration, along with his companion construction reducing any machine to a machine with only two symbols, corroborated the notion derived from experience that machine states can be traded for symbols in such a way as to preserve a roughly constant product of the number of states required times the number of symbols required.

Minsky *1962* demonstrated a four-state by seven-symbol Turing machine to simulate a universal Tag system and in a footnote announced that (he and D. Bobrow had determined) there could not be a two-state by two-symbol Universal Turing Machine. In a private conversation (ca. 1964) Professor John McCarthy of Stanford University remarked, "We thought if we were to find the smallest universal machine then we could learn a great deal about computability—of course that wouldn't be so!" And, the last word representing a prevailing feeling is given by Minsky *1967* who suggests that "the question is an intensely tricky puzzle and has essentially *no* serious mathematical interest."

A related path of investigation has looked at simple possibilities for self-reproduction in systems of cellular automata. Codd *1968* and others continued with the pioneering work of von Neumann who used a cellular automaton model suggested by S.M. Ulam (cf. Burks *1970,* and also Arbib, this volume). Codd's work departed from the usual top-down design of functioning systems by virtue of his use of a computer to search for naturally occurring mechanisms which he could employ in his construction of universal cellular systems.

The size of the smallest universal machine remains unknown, however, whether or not it is of any serious mathematical interest. With our minds remaining open to the possibility of practical, scientific, or simply philosophical interest we shall examine the area of simple Turing machines and systems of cellular automata inspired by two games: Rado's Busy Beaver Game and Conway's Game of Life. We shall look at some questions which are hardly explored and some which may be, for reasons beyond our comprehension, virtually unanswerable.

2. *Turing Machine Questions*

The Busy Beaver Game

The Busy Beaver Game was invented by Tibor Rado *1962.* It is based upon the Kleene *1952* representation of Turing machines in which each sequence of atomic acts consists of printing one symbol and making one move to an adjacent square before changing to the next state. For the Turing machines in his game, Rado added an external *zero* state to denote the act of *halting.* The machines deal with only two symbols, "0" (blank) and "1" (mark). The three-state machine shown in Figure 1 is in the precise form used by Rado as an entry in his game.

		(scanned symbol)		
		0	1	
(current state)	1	1R2	1R0	Score 5
	2	1L2	0R3	Shifts 21
	3	1L3	1L1	

Figure 1. A three-state Turing machine for Rado's Busy Beaver Game.

The Turing machines of the Busy Beaver Game operate on a two-way infinite tape. The tape is initially blank (all zeros), and the "contest" is among machines having the same number of states. The objective is to find the machine which can print the most marks (ones) on its tape before halting. Not all machines halt, of course, so the problem is not merely one of combinatoric unwieldiness, but one of general undecidability. Rado showed that the maximum number of marks which can be placed upon a blank input tape by a machine of k states defines a noncomputable function $\Sigma(k)$. Using essentially a diagonal argument he demonstrated that if f is some computable function then there exists a positive integer n such that

$$\Sigma(k) > f(k) \text{ for all } k > n.$$

Related to the function Σ is another function, the maximum *shift number,* $S(k)$, representing the maximum number of moves or steps that can be taken by a k-state machine which halts after starting on a blank tape. Clearly, $S(k)$ is not computable, else $\Sigma(k)$ could always be computed through a simple process of enumeration once the maximum shift number were known.

The three-state machine shown in Figure 1 will mark five 1's on a blank tape in 21 shifts after beginning in state 1. There are five distinct three-state machines which *score* six, but this is the only machine which will operate for 21 steps (Lin and Rado *1965*). Two machines are shown in Figure 2 which represent those three-state machines (the vast majority) which will never halt. The labels for the two machines describe their classes of nonstopping behavior, and the potential halt entries (unreachable) have been left unspecified.

Experience has shown that a small machine picked at random will very likely never halt. Furthermore, such a machine, started on a blank tape will probably, say, 95% of the time, behave in a trivial way making it obvious

(state) \ (symbol)	0	1
1	1R2	1L3
2	0L1	0R2
3	1L1	---

(a) Counter

(state) \ (symbol)	0	1
1	1R2	0L3
2	1L2	1R1
3	---	1L1

(b) Xmas Tree

Figure 2. Nonstopping three-state Turing machines (blank input tape).

that it will never halt. The machines in Figure 2 (from a set of around one percent of the three-state machines considered for the game) represent a slightly more complicated behavior—not the simple looping displayed by a machine which merely "runs away" in one direction down the length of its tape.

While the halting problem for Turing machines with blank input tapes is recursively unsolvable in general, we have no reason to say that given a particular machine we cannot, for logical reasons, declare that the halting problem for that machine on a blank input tape is unsolvable. (Turing demonstrated that there is a *particular* machine, namely the Universal Machine, for which the halting problem is undecidable for *arbitrary* input.) Notwithstanding the fact that very deep mathematical problems such as Fermat's "Last Theorem" and the Goldbach Conjecture each reduce to deciding the halting problem of an individual Turing machine with a blank input tape, we seem intuitively to relegate the individual problems to the combinatoric realm where an ideal computer with sufficient speed and a large enough memory would provide a solution. Rado, on the other hand, wondered out loud about the possibility of there being a *particular* value of n for which the value of $\Sigma(n)$ could not be decided for *logical reasons,* an untenable position in the eyes of his logician colleagues.

With regards to his Busy Beaver Game, Rado *1963* struck a pessimistic note in declaring that "even though skilled mathematicians and experienced programmers attempted to evaluate $\Sigma(3)$ and $S(3)$, there is no evidence that any known approach will yield the answer, even if we avail ourselves of high-speed computers and elaborate programs. As regards $\Sigma(4)$, $S(4)$, the situation seems to be entirely hopeless at present." His pessimism was premature relative to the value for k. At a time when "high-speed" meant a several microsecond cycle time, and integrated circuits were still in the laboratory, high speed computers were not abundant, and their available time was measured and precious. Nevertheless, the value of $\Sigma(3) = 6$ was soon proved (Lin and Rado *1965*), and the value of $\Sigma(4) = 13$ was discovered and the case for $k = 4$ was reduced to manageable proportions (Brady *1966*).

Still, Rado's assessment was only slightly misplaced, because an unreachable lower bound was subsequently demonstrated for $k > 7$. By means of a recursive construction on the states of a machine Green *1964* was able to put what appears to be a nonprimitive recursive lower bound on $\Sigma(k)$ and show that

$$\Sigma(7) \geq 22,961$$

$$\text{and} \quad \Sigma(8) \geq 3 \cdot (7 \cdot 3^{92} - 1)/2.$$

For a small value of k there was then a score to compare with the age of the universe expressed in nanoseconds! If the problem was then tractable for $k = 4$, where did the real difficulties lie?

In 1971, searching for five-state machines with structures similar to those of Green's, D.S. Lynn *1972* first pushed the lower bound for $\Sigma(5)$ to 22 with 435 shifts. Later, when a large amount of low priority computer time became available, he was able to look at what he estimated to be about ten percent of the *tree-normal* five-state machines and extended the limits to $\Sigma(5) \geq 112$ and $S(5) \geq 7,707$ (Lynn *1974*). There the problem sat until a recent flurry of computer searches by several individuals culminated in a nearly *no contest* entry by Uhing *1986* with new lower bounds of

$$\Sigma(5) \geq 1,915 \quad \text{and} \quad S(5) \geq 2,358,063.$$

Uhing used a microprocessor controlling a "hardware Turing-machine simulator. The hardware simulator was constructed using less than $100 worth of parts and materials, including 32 integrated circuits, sockets and a circuit board." Over several months Uhing simulated about 260,000,000 five-state machines leaving 2,500,000 machines undecided. The high scoring machines are shown in Figure 3.

Even though it might appear now that the five-state problem is within grasp, there is a distinct possibility that the limit of practical solvability has in fact been reached. While we can follow Uhing's current champion machines until they halt, it is not clear at all how the machines work. Any cleverness in their construction is not the result of human creation, so there is a conspicuous absence of documentation! In light of Green's results it was easy to accept that the turning point for the Busy Beaver Game might

	0	1
1	1R2	1L3
2	0L1	0L4
3	1L1	1R0
4	1L2	1R5
5	0R4	0R2

Score = 1,915

Shifts = 2,133,492

	0	1
1	1R2	1R0
2	1L3	1R3
3	0R5	0L4
4	1L3	0L2
5	1R4	1R1

Score = 1,471

Shifts = 2,358,064

Figure 3: High scoring five-state machines discovered by G. Uhing.

occur at $k = 6$, but such magnitudes as have now been produced for $k = 5$ had never been anticipated. Any hope for solving the problem at this level will require computer programs endowed with a level of intelligence that we have not seen in anything done previously by a machine. Can it be decided by a computer program or will it be necessary to assign one mathematician per unresolved five-state Turing Machine?

The Smallest Universal Machine

If a particular Turing machine is a Universal Machine then we know there is no (computable) solution to its halting problem for arbitrary input tapes. With respect to saying whether or not a given machine is universal we can only assert that the machine is *not* universal if the halting question can be decided for any input tape. It essentially requires another machine (i.e., a computer program) which will examine the tape and return the correct "YES" or "NO" answer in every case. But going from blank input tapes to arbitrary input tapes is a giant step. This has not yet been done for the set of two-symbol three-state machines (which, without eliminating trivial cases or any sort of symmetry, number nearly two million). Where in the space of m-symbol by n-state Turing machines might the smallest Universal Machine reside? Perhaps results for the Busy Beaver Game could give us some clues as to where the complex machines lie.

The Busy Beaver Game is naturally open to variations, and the m-symbol by n-state version is one of them. The traditional scoring rule is that only one symbol other than blank must appear on the final tape (Lee *1963*). This is obviously an arbitrary rule, and it may give us no clue as to when a machine stops. If the maximum shift number is known the matter is moot, and it is really the shift number in the end that counts. Some might argue that the range of tape excursion is the true measure of complexity. But, it is not practical to approach the problem from this standpoint, for a machine which will never stop can spend a very long time inside a restricted region. (The three-state *Counter* machine in Figure 2(a) if left running on a 40 square tape will illustrate this point.) So, it will be the shift number in the end which we deal with in deciding whether or not a particular machine will ever halt while we are watching it run.

On the assumption that the shift number represents a kind of *recursive strength* of a machine, we have tabulated values[1] involving $S(m, n)$ for consideration (Figure 4). Along the diagonal running up to the right from (2,5) to (5,2) the author has no values to supply aside from that of Uhing's for $S(5) = S(2, 5)$. The 3×4 and 4×3 machine spaces are apparently orders of magnitude larger than the 2×5 and 5×2 spaces. One might expect to see results at least comparable to that of $S(2, 5)$. The 3×3 space (with "?" entered) appears to be of an order somewhere between the sizes of 2×5 and 2×4. Because it is probably at least an order of magnitude greater than 2×4 (the 2×5 space is nearly three orders of magnitude greater than that of the 2×4) no attempt has been made to supply any value.

		m (symbols)			
		2	3	4	5
n (states)	2	$= 6$	≥ 38	$\geq 7,195$	---
	3	$= 21$	?	---	
	4	$= 107$	---		
	5	$\geq 2,358,064$			

Figure 4. The maximum shift number S for m-symbol by n-state machines.

1 Except for the values for $k = 5$ discovered by Uhing, all lower bound values shown in the paper were produced by computer programs written by the author utilizing a recursive technique involving backtracking to generate distinct machines. While improvements on some of these results no doubt exist, none are known to the author.

The state-symbol product as a rule for conservation of computing power seems to hold across the diagonal from 2×3 to 3×2. There it is possible to say that either $S(3,2) = 38$ or else $S(3,2) > 15,000$, and since there are fewer than 3,000 of the 3×2 machines to deal with it seems safe to conclude that $S(3,2) = 38$. (Someone may well already know this to be a fact—the author does not, but it appears that it should not be too difficult to prove.)

However, what has happened to the conservation rule going across the diagonal where $S(2,4) = 106$ and $S(4,2) \geq 7,195$? (It can be stated that either $S(4,2) = 7,195$ or else $S(4,2) > 15,000$, although in this case settling for the equality does not seem like a safe bet.) There were almost 400,000 machines generated which is about two thirds the number of machines generated for the 2×4 Busy Beaver problem, but seeing a lower bound for the shift number nearly two orders of magnitude larger than the known value of $S(2,4)$ we cannot offhandedly say that the two problems are comparable!

Does a universal machine lie anywhere in this matrix? If the shift number indicating relative recursive strength is a gauge of machine complexity, then it would seem a reasonable guess that universal machines should exist in the machine spaces along the diagonal from (2,5) to (5,2). And what about (4,2)? Why does it appear that there is a significant gain in power from using *four* symbols? Does Nature know this already? The four-symbol by two-state machine space deserves careful scrutiny.

3. *Cellular Automata Problems*

Games in Linear Cell Space

A cellular automata problem equivalent to the Busy Beaver Game was studied by Varshavsky (*1972*) in a linear cell space. An infinite chain of n-state cells all begin in the quiescent state except for one nonquiescent cell which serves as a seed. A cell is taken to be a finite state automaton whose state transition is dependent upon the states of the cells in a three cell neighborhood consisting of itself and the immediately adjacent cells. For fixed value of n Varshavsky wished to determine what is the maximum length L of active cells possible out of which no further growth of activity may occur in the chain. Varshavsky added the rule that once a cell leaves the quiescent state it must remain active thereafter. The states were therefore numbered 0, 1, … n.

This question ranges over all possible transition tables for the cells, a set of functions in number of the order of $n^{(n^3)}$. Varshavsky reported, "By completely enumerating all possible tables of transition rules it has been shown that for $n = 3$, the maximal length $L(3) = 7$. For $n = 4$ transition rules have been found giving $L(n) = 45$ but this length has not been shown to be maximal". No other information was given on how the "enumeration" was performed, nor in particular exactly how it was determined whether or not a chain of cells becomes stable. For $n = 3$ one is dealing with 3^{27} possible transition functions, though in generating the possibilities from a tree of next-state choices and eliminating symmetry in the process one can readily see how the number of possibilities could be greatly reduced.

Vitanyi *1976,* considering Varshavsky's problem, restricted the flow of information in the chain to one direction, and with a definition of what he called a one directional linear cell space (1 LCS), demonstrated that his space defined a Tag system from which it could be deduced that $L(n)$ diagonalizes the computable functions. But, as Vitanyi points out at the end of his paper, the problem is directly "equivalent to the halting problem for Turing machines by encoding the finite control and the scanned symbol in each cell of the linear cell space".

So, how does one reconcile the flow of information in only *one* direction with a construction embedding the encoding of a Turing machine in each cell when a Turing machine requires information to flow in *both* directions? A constructive solution to this is possible. On alternate cycles let the entire state space shift to the right. (Every cell takes on the state of its left neighbor.) Then, a "left" move by the simulated Turing machine can be executed by allowing the active cell location representing the machine's position to "stand still" as the "tape" moves by. A "right" move is executed during a *nonshift* cycle by letting this active cell location make its actual move to the right. In a sense the embedded machines operate in an "expanding universe" of active cells. In his monograph on cellular automata Codd (*1968*) conjectured "that the existence of unbounded but boundable propagation is a necessary condition for computation universality" in a cellular space. One could interpret the simple construction we have presented here as a means for making a demonstration to the contrary.

The problem of determining $L(n)$ for $n = 4$ appears to be out of reach since the possible number of transition functions climbs to 4^{64}. This represents a rather large space to search. On the other hand, a five cell neighborhood with two-state cells has an interesting 2^{32}. Adding Vitanyi's restriction of the cell space to unidirectional information flow, there would be $n^{(n^2)}$ transition functions with a three cell neighborhood, which is only 3^9 for $n = 3$ and the possible 2^{32} for $n = 4$.

Two Dimensions: The Game of Life

The Game of Life devised by J.H. Conway has had such a run of popularity since its introduction in 1970 that it nearly represents a small industry from magazine articles to books to computer software. It was introduced to the public by Martin Gardner in his Mathematical Recreations column in *Scien-*

tific American (cf. Berlekamp, Conway, and Guy *1982* and Gardner *1983*). It has stimulated both philosophical commentary and science fiction and possesses an entertaining and virtually unending taxonomy of cellular phenomena, containing such combinations as "blinkers", "beehives", "boats", "pulsars", "gliders", and even a "Cheshire cat".

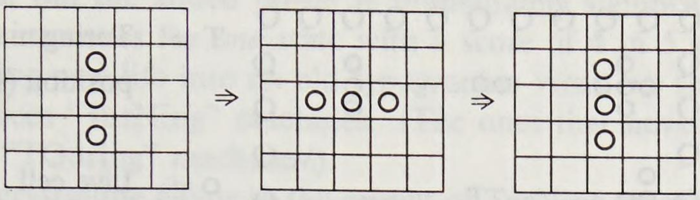
The rules for the game were devised after a certain amount of experimentation. It has a fascinating biological naturalness. Two-state cells ("dead" or "alive") occupy an infinite two-dimensional rectangular space in which only a finite number of cells are initially in the living state. A birth (transition from dead to alive) occurs in a cell at the center of nine squares only if exactly three live neighbors are present. Isolated living cells with no more than one live neighbor will die, and crowded cells with four or more live neighbors will die. Living cells with two or three neighbors will survive.

Conway studied many initial configurations and originally thought that all finite initial patterns eventually degenerated into stable or oscillating patterns (Figure 5) or else disappeared entirely. He offered a $50 prize for anyone finding a pattern which would grow without bound. Such patterns were discovered and led to the creation of an entire system of patterns which could be used to simulate a computer. In other words, with a particular encoding of its cell space, Life becomes a Universal Turing Machine!

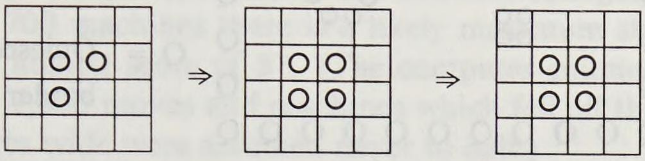
In the nine cell Life neighborhood there are 2^{512} possible transition functions (encompassing as well all 2^8 possible neighborhoods involving immediately adjacent cells). Can Conway's Life be the only interesting function? For example, admitting anisotropic transition functions leads to the possibility of directly simulating simple neural nets. Naturally, Life already supports neural nets as a programming layer on top of its simulated computing machine, but the issue here is examination of simple mechanisms functioning directly in a cellular space.

What about the two-dimensional equivalent to Varshavsky's $L(n)$? We have pointed out that the transition space for $n = 2$ has 2^{512} possibilities. It is difficult to imagine the problem ever being solved for $n = 2$, at least not by any exhaustive enumeration. It would seem to require a theoretical solution.

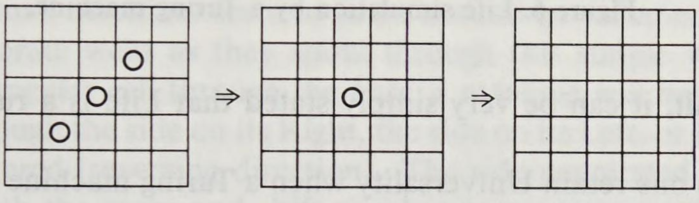
A cellular universe as an infinite collection of active devices lacks the intuitive appeal of a Turing machine with its infinitely extensible passive medium. However, by restricting the nonquiescent cells to a finite number the two systems are seen to be equivalent. Still, in a very small region of cells with a only a few states little is known which enables us to predict their behavior. These frontiers of Life have barely been explored. It is easy to view anyone's interest in this area as a frivolous preoccupation, because it is difficult to admit that the problems in a simple and seemingly natural domain vastly exceed our present mathematical understanding.



(a) Blinker (oscillating)



(b) Block (stable)



(c) Diagonal chain (vanishing)

Figure 5. Simple cellular examples in Conway's Game of Life.

4. The Busy-Beaver Game in Two Dimensions

Simulation of Life

If a Turing machine is able to operate in two-dimensions (on a rectangular grid) it is a straightforward matter to design one to simulate the Game of Life. For instance, position the Turing machine in the exact center of a large square border surrounding the active Life cells (Figure 6) where it proceeds in a spiral fashion from the center toward the border leaving a trail as it moves. On the outward pass the Turing machine temporarily marks all those cells which will give birth and also marks all the cells which will die. If any cells give birth while the machine moves around the quiescent cells comprising the border itself, the Turing machine expands the border in all directions by one layer of cells. After completing its trip out to the border, the machine spirals back toward the center marking the new cells as living, deleting the dead cells, and cleaning up its trail. From the center the Turing machine repeats the process for the next cell transition.

Our Turing machine equivalent to the usual Life simulating computer program accommodates Life's hypothetical infinite grid. Since this machine

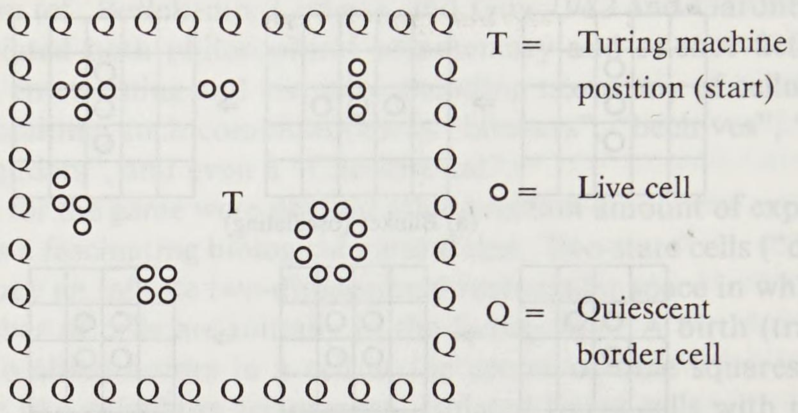

T = Turing machine position (start)
o = Live cell
Q = Quiescent border cell

Figure 6. Life simulation by a Turing machine.

will never halt, it can be very simply stated that Life is a runaway Turing machine!

How does one retain Universality when a Turing machine is constrained never to halt? The simple Turing machine just described is a Universal Machine. It simulates Life, and Life is Universal. The matter of halting comes directly from Conway's original question regarding the stability of Life configurations. We equivalently ask whether or not the border will ever stop expanding. This entire described system (infinite Life grid and simulating Turing machine) can itself be simulated by a Turing machine on its own one-dimensional infinite tape. So the question of border expansion can be tied to the halting problem for the machine with the tape: it merely stops if it has to wait too long for the border to expand. This is not practical, but it is theoretically possible since having enough time to wait is no more of a problem for a Turing machine than having enough tape.

"TurNing" Machines

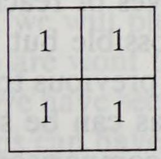
Moving Turing machines into the two-dimensional domain of Life does nothing to enhance their ultimate computational power. Turing reduced the computing problem from two dimensions to one dimension in the first place in order to get at the simple essence of computer mechanism. However, if we limit ourselves to small machines of approximately the same size, it is reasonable to wonder what degree of recursive strength might be added by removing the one-dimension restriction.

Rather than add the obvious extensions of "Up" and "Down" to "Left" and "Right" why not allow a machine to *turn* as well as move? Let *Left, Right,* and *Back* reorient the machine at the same time it moves, while *Forward* takes it straight ahead. This sacrifices a sense of direction in the external world, but the added power is immediately significant: the Busy Beaver Game improves for *one* state with a score of 4 in 5 shifts. At the risk of breathing new life into an old typographic virus, we shall call these enhanced devices "TurNing" machines. (The ones that never stop can be referred to as "TOuring" machines!)

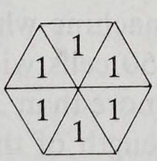
There is a crystalline flavor in the output of TurNing machines with two states (as one might suspect), and over the machine configuration space of little more than 700 machines there is a likely maximum shift number of 121 and a likely highest score of 37. (The computer enumeration process was restricted to 2,000 moves and machines which fell off the "edge of the earth" 100 squares wide were assumed never to halt.)

An interesting simplification of these machines is obtained by restricting their action to a triangular grid. In the sort of *Chinese checkers* space which results, very small machines seem to find a natural environment where they can spin elaborate webs as they spiral through this simple world. Upon entering a triangular cell through the *base,* a machine will have the choice of moving through the side on its Right, the side on its Left, or Back through the side it entered (reversing direction). The side penetrated becomes the new base. Both the score and shift number improve immediately: a one state machine can turn through six cells! (See Figure 7.)

	(symbol)	
	0	1
(state) 1	1 R 1	1 R 0



Score = 4
Shifts = 5
(square cells)

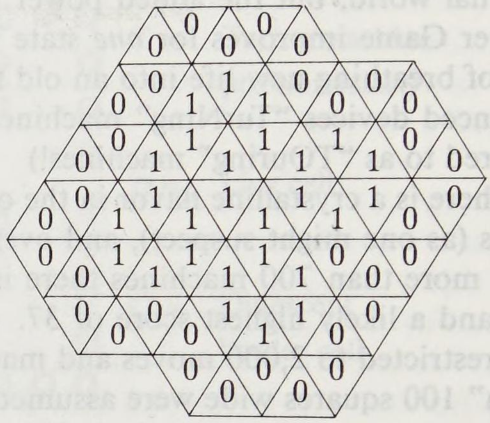


Score = 6
Shifts = 6
(triangular cells)

Figure 7. A one-state "TurNing" machine.

Among 356 two-state triangular machines examined (which should exhaust the possible configurations) the maximum shift number found was 171, achieved by a machine which also visited the most cells, a total of 62 (Figure 8). A different machine achieved the highest score of 39. If these are not the maximum values then the shift number probably exceeds 2,000.

(state) \ (symbol)	0	1
1	1 R 1	1 R 0
2	0 L 2	1 B 1



Shifts = 171, Cells visited = 62, Score = 27

Figure 8. A high scoring two-state machine in two dimensions.

The number of distinct three-state machines for the game remains quite manageable. Among more than 91,000 machines of the triangular version inspected the highest shift number found was 1,721 and the highest score encountered was 351. Since no machines were watched beyond 2,000 moves, it is not reasonable to predict that either the maximum shift number or the maximum score has been found. It is entirely possible that there are more machines to be inspected, for the generation tree was trimmed on the assumption that any machine which continued past 2,000 moves or reached the edge of a space 50 cells wide did not halt. (It is also possible but not likely that there are more than 356 two-state machines in the previous tree).

The recursive strength of these machines grows rapidly as can be seen from the results of computer runs tabulated in Figure 9. A comparison of the shift number values among the various cases in both one and two dimensions suggests that an attempt to solve the halting problems here would encounter a much higher degree of difficulty than that handled by Brady *1983* for a final proof that $\Sigma(4) = 13$. At least the number of machines to handle would not be unreasonable.

The effect of the addition of more symbols to two-dimensional machines was not examined. A four-symbol by two-state TurNing Machine might well surprise us in its complexity!

	Triangle cells		Square cells	
	$S(k)$	$\Sigma(k)$	$S(k)$	$\Sigma(k)$
states $k = 1$	$= 7$	$= 6$	$= 5$	$= 4$
2	≥ 171	≥ 39	≥ 121	≥ 37
3	$\geq 1,721$	≥ 351	?	?

Figure 9. Busy Beaver results in two dimensions: "TurNing" machines.

5. *Conclusion*

To the more pragmatically minded the Universal Computing Machine is represented on a silicon slate using nor-gates for chalk. The minimal machine in this view is a single nor-gate. It is perhaps just another way of saying that as the number of states is reduced the complexity of the encoding is increased, and in an infinite gate array there is really no distinction between the finite-state machine and its tape. This is not a satisfying view, however. It begs the question.

At another level in the practical world we would experiment with confidence on the 25,000 node neural net representing a snail's brain, or snip off a piece of a chromosome and admire the effect on the offspring of the altered cell. We do this without admitting the limitations of our knowledge of the underlying mechanisms. It is not to say that experimentation is wrong—on the contrary, we have emphasized a problem area involving simple mechanisms wherein we would strongly encourage experimentation. But, we will plead at the same time for more humility on the part of those who are wont to experiment at the high range of the scale.

We have seen how some relatively simple variations in very small mechanisms can have disproportionate and possibly unanticipated effects on their recursive power. How this comes about is not at all clear. Expansion (adding states or symbols) ultimately yields universality in a form which we are able to program in ways consistent with our experience. So, in a strict logical sense, the changes offer nothing new. Nature, however, may have its own methods of programming. Not only should we study the automata mentioned here, but we should be considering many other variations as well. The experience gained could lead to the practical discovery of parallel mechanisms active in natural phenomena. And, furthermore, the mechanized proofs required for rigorous solutions to these problems will give us a sense of the degree to which we can honestly and realistically apply the modifier *artificial* to the term *intelligence.*[2]

Our concluding perspective will be summarized in the following conjectures and predictions proceeding from machines with just one state to machines with six.

Conjecture 1. There exists a *one-state* Universal Machine which operates in two dimensions (i.e., a one-state "TuRning" machine).

Likelihood that this conjecture will be proved: Fair – Nature has probably already proved it, but with time on its side.

Conjecture 2. There does not exist a two-symbol by two-state Universal Machine if *halt* entries are excluded.

Likelihood that this conjecture will be proved: Good, but the characterization of "looping" as a substitution for halting may be an open-ended matter. Halting is useful to make a theoretical point, but in real life survival is the tautological goal. And we no longer build computers which halt—at most they wait in a quiescent state. In any case, solving this problem would be good practice for attacking the following.

Conjecture 3. The halting problem for two-symbol by three-state Turing machines is decidable.

Likelihood that this conjecture will ever be proved: Fair, but this is an extremely challenging problem. Let anyone in doubt pick some machines and try it!

Conjecture 4. There exists a two-symbol by four-state Universal Machine if *halt* entries are excluded. (This is prompted by intuition bolstered by the fact that with four states it is possible to send two signals in each of two directions, and by excluding halt entries some of the recursive strength of the five-state space is assumed.)

Likelihood that this conjecture will ever be disproved: Nil.

Prediction 5. It will never be proved that $\Sigma(5) = 1,915$ and $S(5) = 2,358,064$. (Or, if any larger lower bounds are ever found, the new values may be substituted into the prediction.)

Reason: Nature has probably embedded among the five-state holdout machines one or more problems as illusive as the *Goldbach Conjecture.* Or, in other terms, there will likely be nonstopping recursive patterns which are beyond our powers of recognition.

Prediction 6. From known results for $k = 5$ a six-state machine will be constructed for which it can be "proved" that its shift number (and thus a lower bound for $S(6)$) is an incomprehensibly large value which is in itself difficult to describe.

2 House and Rado *1963* proposed the study of small Turing machines as a relevant endeavor for the then infant specialty of *artificial intelligence.* They pointed out the theoretical and practical difficulties for problems such as automated searches for optimal machines.

Reason: It is now clear that determining $\Sigma(6)$ and $S(6)$ is intractable. At this level one can speculate with impunity, and we shall.

Some students of the author were readily convinced after extensive examination and computer testing that Uhing's champion machine for $S(5)$ would never halt. Seeking assurance one student ran her simulator to a point just short of two million moves! From an amusing experience such as this, one is led to consider the possibility that someday a machine of six states (or a just a few more) will be presented by a group of mathematicians along with a "proof" that it will never halt. Suppose then an efficient simulator for the machine were built on the leading but slightly jagged edge of technology and run for an extensive period of time. And then suppose it were to halt! The mathematicians, with solid reasoning to back them up, could make a valid claim that the machine malfunctioned.

But now suppose that instead of building a machine, another group of equivalently qualified thinkers, supported by a great body of mathematical knowledge, countered with a different "proof" that after some unimaginable number of moves the proposed machine would in fact halt. Their number would be so large that building a simulator to check the result would be inconceivable. What then? (It is only speculation, of course!)

For $k = 6$ the problem transcends mechanism. One reaches a point where it becomes impossible to distinguish between the finite and the infinite. Is there a point at which it will transcend logic? Radó's question remains open.

Acknowledgements. The author wishes to thank Luanne Napoli for assistance in editing, Ashraf Kaiser for producing the figures, the American University in Cairo for the use of computer laboratory facilities, and the editor of this volume for his encouragement and patience.

References

Arbib, M.A.

1977 From Universal Turing Machines to Self-Reproduction, this volume.

Berlekamp, E., J. Conway, and R. Guy

1982 *Winning Ways for Your Mathematical Plays.* Orlando: Academic Press.

Brady, A.H.

1966 The conjectured highest scoring machines for Radó's $\Sigma(k)$ for the value $k = 4$. *IEEETEC, EC-15* **5** (Oct. 1966) 802–803

1983 The determination of the value of Radó's noncomputable function $\Sigma(k)$ for four-state Turing machines. *Math. Comput.* **40** (April 1983).

Burks, A.W. (ed.)

1970 *Essays on Cellular Automata.* Urbana: University of Illinois Press (1970).

Codd, E.F.

1968 *Cellular Automata.* Orlando: Academic Press (1968).

Gardner, M.

1983 *Wheels, Life, and Other Mathematical Amusements.* New York: W.H. Freeman (1983).

Green, M.W.

1964 A lower bound on Rado's sigma function for binary Turing machines. In: *Proceedings of the 5th Annual IEEE Symposium on Switching Circuit Theory and Logical Design,* pp. 91–94 (Nov. 1964).

House, R. W., and T. Rado

1963 An approach to artificial intelligence. IEEE Special Publication S-142 (Jan. 1963).

Kleene, S.C.

1952 *Metamathematics.* Princeton: Van Nostrand (1952).

Lee, C.Y.

1963 Lecture Notes from the University of Michigan Engineering Summer Conference, Ann Arbor (1963) 18–21.

Lin, S., and T. Rado

1965 Computer studies of Turing machine problems. *J. ACM* **12** (April 1965).

Lynn, D.S.

1972 New results for Rado's sigma function for binary Turing machines. *IEEETEC C-21***8** (Aug.1972).

1974 Private communication.

Minsky, M.

1962 Size and structure of universal Turing machines using Tag Systems, Recursive Function Theory. *Sym. P. Math.* **5** AMS (1962).

1967 Computation: Finite and Infinite Machines. Englewood Cliffs, NJ: Prentice-Hall (1967).

Rado, T.

1962 On non-computable functions. *Bell Sys. T.* (May 1962) 877–884.

1963 On a simple source for non-computable functions. In: *Proceedings of the Symposium on Mathematical Theory of Automata, Polytechnic Institute of Brooklyn,* pp. 75–81 (April 1963).

Shannon, C.E.

1956 A universal Turing machine with two internal states. In: *Automata Studies,* eds. C. Shannon and J. McCarthy. Princeton: Annals of Math. Studies (1956).

Turing, A.M.

1936-7 On computable numbers with an application to the Entscheidungsproblem. *P. Lond. Math. Soc. (2)* **42** (1936-7) 230–265.

Varshavsky, V.I.

1972 Some effects in the collective behavior of automata. *Mach. Intell.* 7 (1972) 389–403.

Vitanyi, P.M.B.

1976 On a problem in the collective behavior of automata. *Discr. Math.* **14** (1976) 99–101.

Uhing, G.

1986 Unpublished notes, dated Feb. 4. (also in A.K. Dewdney, Mathematical recreations, *Sci. Am.* **252** (April 1985) 20–30.)

[illegible], F.F.

Gardner, M.

1983 *Wheels, Life, and Other Mathematical Amusements.* New York: W.H. Freeman (1983).

Green, M.W.

1964 A lower bound on Rado's sigma function for binary Turing machines. In: *Proceedings of the 5th Annual IEEE Symposium on Switching Circuit Theory and Logical Design,* pp. [illegible] (1964).

House, R. W., and T. Rado

1963 An approach to artificial intelligence. IEEE Special Publication S-142 (Jan. 1963).

Kleene, S.C.

1952 *Metamathematics.* Princeton: Van Nostrand (1952).

Lee, C.Y.

1963 Lecture Notes from the University of Michigan Engineering Summer Conference, Ann Arbor (1963) [illegible].

Lin, S., and T. Rado

1965 Computer studies of Turing machine problems. *J. ACM* 12 (April 1965).

Lynn, D.S.

1972 New results for Rado's sigma function for binary Turing machines. *IEEE TEC* [illegible] (Jan. 1972).

1974 Private communication.

Minsky, M.

1962 Size and structure of universal Turing machines using Tag Systems, Recursive Function Theory, *Sym. P. Math.* 5 AMS (1962).

1967 Computation: Finite and Infinite Machines. Englewood Cliffs, NJ: Prentice-Hall (1967).

Rado, T.

1962 On non-computable functions. *Bell Sys. T.* (May 1962) 877–884.

1963 On a simple source for non-computable functions. In: *Proceedings of the Symposium on Mathematical Theory of Automata,* Polytechnic Institute of Brooklyn, pp. 75–81 (April 1963).