Cellular Automaton Supercomputing

Stephen Wolfram
Center for Complex Systems Research,
and Departments of Physics, Mathematics and Computer Science,
University of Illinois at Urbana-Champaign, 508 South Sixth Street,
Champaign, IL 61820.

Many of the models now used in science and engineering are over a century old. And most of them can be implemented on modern digital computers only with considerable difficulty. This paper discusses some new basic models which are much more directly suitable for digital computer simulation.

The ultimate purpose of most scientific investigations is to determine how physical or other systems will behave in particular circumstances. Over the last few years, computer simulation has been emerging as the most effective method in many different cases. The basic approach is to use an algorithm which operates on data in the computer so as to emulate the behaviour of the system studied (e.g. [1]). This algorithm can be considered to provide a ''computational model'' for the system.

Theoretical investigations of physical systems have conventionally been based on a few definite classes of mathematical models. By far the most common are partial differential equations (e.g. [2]). These equations were designed to describe systems such as fluids which can be considered as continuous media. Calculus was used as a tool to find mathematical formulae for the solutions to these equations. This allowed great progress to be made in the understanding of many phenomena, particularly those such as electromagnetism, which are by linear partial differential equations. Progress was also made in studies of processes such as laminar (regular) fluid flows, which can be approximated by linear partial differential equations. But the standard methods of mathematical analysis made little headway on problems such as fluid turbulence, for which non-linear partial differential equations are essential.

When digital computers became available, it was natural that they should be used to try and find solutions to such partial differential equations. But digital computers can represent such equations only approximately. While equations involve continuous variables, digital computers can treat only discrete, digital, quantities. The real numbers which correspond to continuous variables in the equations must be represented on the computer by packets of bits, typically in the form of 32 or 64 bit numbers in floating-point format. In addition, the derivates which appear in the equations must be approximated by finite differences on a discrete grid. Much effort has been spent in numerical analysis to show for example that with sufficiently fine grids, exact solutions to the continuum equations can be found. Unfortunately, such theorems have been proved almost exclusively only in cases where exact solutions to the continuum equations are known. For most important non-linear equations, quite ad hoc methods must be used to gauge the accuracy of approximations.

Nevertheless, the thrust in scientific computation has been to develop computer hardware and algorithms which allow more and more extensive approximations to partial differential equations to be made. Thus, for example, the performance of computers is often measured in terms of the rate at which they can carry out the floating point operations needed. In many cases, there seem to be limitations which will prevent rapid increases in such performance.

Significant progress may perhaps more easily be made by somewhat shifting the emphasis. The kinds of operations which can efficiently be carried out by digital electronic circuits, and thus digital computers, are quite clear. Large numbers of simple logical operations can be performed, potentially in parallel on many elements of a regular grid. Given the structure, one may then ask the question of whether accurate computational models based on this structure can be found for physical and other systems.

Cellular automata (e.g. [3,4]) provide one class of examples. A cellular automaton consists of a discrete lattice of sites. Each site carries a discrete value, chosen from a small set of possibilities. The values are updated in a sequence of discrete time steps, according to logical rules which depend on the values of neighbouring sites. Cellular automata are thus, by construction, almost ideal for simulation on digital electronic computers. They are particularly well suited for the coming generation of massively-parallel machines, such as the Connection Machine computer [5], in which a very large number (currently 65536) of separate processors, each simple, act in parallel.

One of the most remarkable results of recent studies on cellular automata is that even with very simple rules, it is possible to obtain behaviour of considerable complexity [3,4]. Figure 1 shows a few examples. The rules consist of just a few simple logical operations. But when they are applied over and over again, their collective effect can yield very complex patterns of behaviour. Often these show striking similarities to forms seen in many natural systems, and in other mathematical models for these systems. Chaotic behaviour, corresponding to strange attractors, is common in cellular automata. Fractal patterns are also, for example, often produced.

One thus expects that very simple computational models, based say on cellular automata, should suffice to reproduce many different natural phenomena. The challenge is to abstract the essential mathematical features of the phenomena, so as to be able to capture them in as simple a model as possible.

As one example, I shall discuss here some recent models for fluid flow phenomena, based on cellular automata (e.g. [6]).

Fluids are conventionally described by the Navier-Stokes partial differential equations (e.g. [7]). These equations can presumably in principle describe the important phenomenon of fluid turbulence. But digital computer simulations based on the Navier-Stokes equations are barely able to reach the regime needed to reproduce turbulence accurately. Of course, the Navier-Stokes equations are themselves an approximation. At a fundamental level, fluids consist of discrete particles, usually molecules. The Navier-Stokes equations give an approximate continuum description of the average behaviour of large numbers of such discrete particles. When the Navier-Stokes are simulated on digital computers, however, discrete approximations must again be made. These approximations, perhaps in the form of finite differences, bear little resemblance to the original system of discrete particles. Yet in the limit of a large number of discrete elements, they too should correspond to the continuum Navier-Stokes equations.

A wide variety of systems, with very different microscopic dynamics, in fact appear to follow the Navier-Stokes equations in the large-scale limit. Thus, for example, air and water, despite their very different molecular constitution, can both be described by the Navier-Stokes equations, albeit with different values of parameters such as viscosity.

In an attempt to devise the most efficient computational models for fluids, one may try to find the simplest microscopic dynamics which reproduces the Navier-Stokes equations in the macroscopic limit. Such models may correspond to optimal algorithms for determining the behaviour of a fluid using a digital computer.

One class of computational models is based on a simple discrete idealization of molecular dynamics [6]. Particles move in discrete steps along the links of a fixed lattice, with each link supporting say at most one particle. The particles collide and scatter according to simple logical rules. The rules are arranged so as to conserve the total number of particles, and the total momentum carried by these particles. Fluid behaviour can potentially be obtained in this system by considering the values of bulk quantities such as particle density or momentum density, averaged over a large lattice regions.

Figure 2 shows some results obtained in this way. Detailed studies have demonstrated that many of the phenomena seen in actual fluid experiments can accurately be reproduced by this simple cellular automaton model. Figure 2 shows calculations of two-dimensional flow past a cylinder. The standard transition from steady flow to a regular vortex street is observed. Then at higher Reynolds numbers (dimensionless fluid flow rates) the vortex street is seen to become aperiodic, corresponding to the onset of turbulent behaviour.

The cellular automaton method used in figure 2 may well be practical for many fluid dynamics computations. Through its close correspondence with the underlying physics of fluids, it is straightforward to include many physical effects and constraints. Thus for example solid objects with arbitrary shapes, and possibly, say, flexible boundaries, can easily be treated. In our current implementation on a Connection Machine computer with 65536 processors, lattices of size say 4096×8192 can be updated at a rate of about 10^9 sites per second, allowing the fluid flow patterns around objects to be found interactively up to Reynolds numbers of several hundred. The readily scalable architecture of the Connection Machine computer makes much larger simulations with the same method quite feasible in the future.

At a theoretical level, cellular automaton fluid models can be analysed by much the same methods of statistical mechanics as have been used in trying to derive the Navier-Stokes equations for physical fluids from the microscopic dynamics of real molecules. One approach is to use kinetic theory to derive transport equations for the average densities of particles with particular positions and directions (e.g. [8]). In the hydrodynamic limit, these microscopic average densities can be approximated through a Chapman-Enskog expansion in terms of macroscopic fluid densities and velocities. The resulting equations for these macroscopic quantities correspond closely with the usual Navier-Stokes equations. Just like a real fluid, however, the cellular automaton model contains definite higher-order corrections, not included in the Navier-Stokes equations. In addition, analytical methods provide only approximate values for parameters such as viscosity; accurate values must be obtained from explicit computer simulations.

A fundamental assumption of the kinetic theory method is that the microscopic configurations of particles can be specified purely in terms of probabilities, which are in turn determined by the values of averaged quantities. This is essentially equivalent to the assumption of thermodynamic equilibrium, and is related to the fundamental principles of thermodynamics.

The Second Law of thermodynamics suggests that even if the initial configuration of particles is orderly, it will become progressively more disordered as a result of the motion and collisions of particles, and will show for example an increasing coarse-grained entropy. This phenomenon occurs if the evolution of the cellular automaton, even from "simple" initial conditions, yields behaviour that is so complicated as to seem random for practical purposes.

Very simple examples of cellular automata are known in which such apparent randomness can be produced. Figure 3 shows a one-dimensional example [9,10]. Even starting from an initial state containing a single nonzero site, many features of the pattern produced, such as the sequence of values in the center vertical column, are sufficiently random that they pass standard statistical tests of randomness [9]. The cellular automaton evolution thus acts like a pseudorandom number generator: even though a simple seed is given, the algorithm yields sequences whose simple origins cannot be discerned. The evolution of the system thus effectively "encrypts" the initial data: given just the output sequence it is very difficult to deduce the original seed. The cellular automaton of figure 3 can in fact be used as an efficient practical random sequence generator or stream encryption algorithm [11] (it is for example the primary pseudorandom generator used on the Connection Machine computer).

There are many mathematical systems which act in this way. It is for example easy to specify π, or to generate its digits. Yet once generated, the sequence of digits seems random for all practical purposes. Observations of this kind are related to the general conjecture of computational complexity theory (e.g. [12]) that $P \neq NP$. Computations which can be performed in polynomial time (P) seem to have inverses (which must be in the class NP) that require more than polynomial time, and probably often correspond to computations that are infeasible in practice.

Many mathematical models of physical processes probably show such behaviour [10]. Even with simple initial data, they rapidly yield configurations which seem random for practical purposes. Such behaviour may well be the basis for the widespread validity of the Second Law of thermodynamics. One of its important consequences is that a probabilistic or statistical description should indeed be valid for many systems such as cellular automaton fluid models. Such a description would depend only on macroscopic average variables. This may explain why different microscopic models often yield the same macroscopic behaviour. It is basic reason that simple discrete dynamics can give essentially the same overall behaviour as the full dynamics of physical molecules.

Statistical descriptions of cellular automaton fluid models are close in form to explicit finite difference approximations to partial differential equations. In both cases, each site on a grid carries a continuous variable which describes the average density and velocity of the fluid at that point. In practical computations with the finite difference method, this variable is typically represented directly as a floating-point number. In the cellular automaton method, the variable can be viewed as represented in a probabilistic or statistical fashion.

Following the usual development of statistical mechanics, a statistical description of a cellular automaton fluid can be obtained as an average over an ensemble of possible microscopic particle configurations. But an actual cellular automaton fluid simulation involves the evolution of just a single, specific, microscopic configuration. Nevertheless, following a fundamental assumption of statistical mechanics, one expects that suitable space or time averages of this specific configuration should yield results which are close to those obtained from averages over the whole ensemble.

This interpretation allows a comparison between cellular automata and discrete approximations to partial differential equations. In the latter case, ensemble average properties are considered, and their evolution is followed precisely. In the former case, just a single instance of the ensemble is considered, and macroscopic quantities are obtained as explicit averages over microscopic variables. If the fundamental assumptions of statistical mechanics are indeed valid, one expects that the cellular automaton method cannot fail to be more efficient than the finite difference one. For much of the information manipulated in the finite difference case is undoubtedly irrelevant to the macroscopic behaviour of interest.

Some evidence for this comes from the fact that most fluid computations yield results which are accurate to at most the percent level. Yet in the finite difference approach, fluid velocities at individual grid points are typically stored to 16-decimal-digit accuracy. Presumably it is only the most significant few digits, and certain overall features of less significant digits, which affect the final results. In the cellular automaton method, all bits of information about microscopic particle configurations are equally important. The cellular automaton representation may thus be a more efficient encoding of the state of the fluid.

The cellular automaton approach to fluid dynamics is but one example of an expanding set of computational models which are based on the collective properties of large numbers of simple discrete components. Standard cellular automata with deterministic rules have been used as models for reaction-diffusion systems, dendritic growth processes, dynamic spin systems, aggregation processes, and many other phenomena (e.g. [3]). Intrinsically probabilistic rules can also be used, and their consequences deduced by Monte Carlo sampling. The resulting models have been used extensively in studying quantum fields and many other systems.

In practice the probabilistic elements of such models must be implemented on digital computers using pseudorandom number generation algorithms. The resulting complete computational model, including the pseudorandom number generator, must thus be entirely deterministic. And since even very simple deterministic cellular automata can yield a high degree of randomness, one expects that formally probabilistic models can be replaced by deterministic ones, often involving a smaller total number of steps. One example of this occurs for the Ising spin system model, which is conventionally studied by updating spins probabilistically, but for which a more efficient algorithm based on a simple deterministic cellular automaton is known [13].

In general, there may be many different cellular automaton models for any particular system. Although the microscopic rules are different, their large-scale or continuum behaviour may be equivalent. In seeking the most efficient simulation algorithm for a particular system, one must find the ''simplest'' cellular automaton rules which yield the required large-scale behaviour.

Most computational models are created by explicit construction. Like most computer programs, each step or feature of their construction is specifically designed to have particular, known, consequences. But in most cases, this methodology will not yield truly optimal programs. Instead, one may imagine defining particular goals or constraints, and then searching the space of possible programs for the optimal ones which achieve these goals (e.g. [14,15]). This approach is particularly promising for problems such as finding optimal cellular automaton rules, in which the space of possible programs has a comparatively simple structure. Thus for example one may consider searching for the simplest cellular automaton rule which has a particular form of large-scale behaviour. Typically the space of possible rules can be reduced by imposing certain constraints, such as microscopic conservation laws, but the suitability of any particular rule can usually be determined essentially only by explicit simulation. The randomness-generating rule of figure 3 was found by such a search-based method.

The problem of finding optimal cellular automaton rules is in many ways analogous to problems such as the optimization of Boolean logic circuits, or the layout of large-scale integrated circuits. The overall goal is defined by the function to be implemented, but the most efficient circuits or rules can usually not be obtained by explicit construction. Instead one searches a large number of candidates, typically using a computer, and finds which of them is best.

Rather than performing an exhaustive search of possible circuits or rules, it is often better to use an iterative or adaptive procedure. One begins with a particular circuit or rule which has been constructed to satisfy the constraints that have been imposed. Then one makes a sequence of ''moves'' in the space of possible rules or circuits, with each move arranged so that the constraints are still satisfied. In the simplest cases, each move is chosen to yield a circuit or rule which is more optimal, or may be considered to have a lower ''cost''. But such a ''gradient descent'' method can find optima only when the ''landscape'' associated with the problem (whose height gives the cost for a circuit represented by a particular point) is essentially a smooth bowl. For many actual problems, the landscape seems closer to a ''mountainous'' or fractal one, on which the gradient descent method will get stuck in local optima. Simulated annealing seems to be a more promising general technique for optimization in such cases [16]. With this method, randomness is introduced into the choice of moves. Initially, a high level of randomness is used, so that the moves are sensitive only to the gross features of the landscape. The randomness is progressively decreased, so that optimization is carried out with respect to smaller and smaller scale features of the landscape.

As one example of such ''adpative programming'', I have recently been searching for the simplest one-dimensional cellular automaton rule which reproduces the diffusion equation in the large scale limit. (For another example, see ref. [17]). The rule must conserve a scalar additive quantity (analogous to particle number), but must generate randomness on a microscopic scale. In addition, the rules were chosen to be microscopically reversible, so that, by analogous with real physical systems, the evolution of the system can be uniquely reversed. Figure 4 shows the behaviour of a rule found by a search over a particular class of simple rules. Starting from a simple initial state, the rule generates progressively more random microscopic configurations. Although the simple initial conditions can in principle be recovered at any time by reversing the evolution, it becomes progressively more difficult to do so. As discussed above, this phenomenon may well illustrate the fundamental basis for the Second Law of thermodynamics. With the rule of figure 4, macroscopic average densities should follow the diffusion equation. As a result, slow spatial variations in density are for example damped on average according to the diffusion equation.

This paper has discussed some new directions for computational modelling. The fundamental principle is that the models considered should be as suitable as possible for implementation on digital computers. It is then a matter of scientific analysis to determine whether such models can reproduce the behaviour seen in physical and other systems. Such analysis has now been carried out in several cases, and the results are very encouraging.

References

1. S. Wolfram, ''Computer software in science and mathematics'', Scientific American (September 1984).
2. A. Sommerfeld, *Partial differential equations of mathematical physics*, (Academic Press, 1955).
3. S. Wolfram (ed.), *Theory and applications of cellular automata*, (World Scientific, Singapore, 1986).
4. S. Wolfram, ''Cellular automata as models of complexity'', Nature 311 (1984) 419.
5. W. D. Hillis, *The Connection Machine*, (MIT Press, 1985).
6. U. Frisch, B. Hasslacher and Y. Pomeau, ''Lattice gas automata for the Navier-Stokes equation'', Physical Review Letters 56 (1986) 1505 [reprinted in ref. 2]; J. Salem and S. Wolfram, ''Thermodynamics and hydrodynamics with cellular automata'', in ref. 2; S. Wolfram, ''Cellular automaton fluids 1: Basic theory'', Journal of Statistical Physics, (to appear, November 1986); B. Nemnich and S. Wolfram, ''Cellular automaton fluids 2: Basic phenomenology'', Center for Complex Systems Research/Thinking Machines Corporation report, to appear.
7. D. J. Tritton, *Physical fluid dynamics*, (Van Nostrand, 1977).
8. A. Sommerfeld, *Thermodynamics and statistical mechanics*, (Academic Press, 1955).
9. S. Wolfram, ''Random sequence generation by cellular automata'', Advances in Applied Mathematics 7 (1986) 123.
10. S. Wolfram, ''Origins of randomness in physical systems'', Physical Review Letters 55 (1985) 449.
11. S. Wolfram, ''Cryptography with cellular automata'', in Proceedings of CRYPTO '85 (Santa Barbara, August 1985).
12. J. Hopcroft and J. Ullman, *Introduction to automata theory, languages, and computation'', (Addison-Wesley, 1979).*
13. M. Creutz, ''Deterministic Ising dynamics'', Annals of Physics 167 (1986) 62; Y. Pomeau, ''Invariant in cellular automata'', Journal of Physics A17 (1984) L415.
14. D. Farmer, A. Lapedes, N. Packard and B. Wendroff (eds.), *Evolution, games and learning*, Physica D, to appear.
15. S. Wolfram, ''Approaches to complexity engineering'', in ref. 2, and to appear in ref. 14.
16. S. Kirkpatrick et al., ''Global optimization by simulated annealing'', Science 220 (1983) 671.
17. N. Packard, ''Adaptation in the space of cellular automaton rules'', Center for Complex Systems Research report, in preparation.

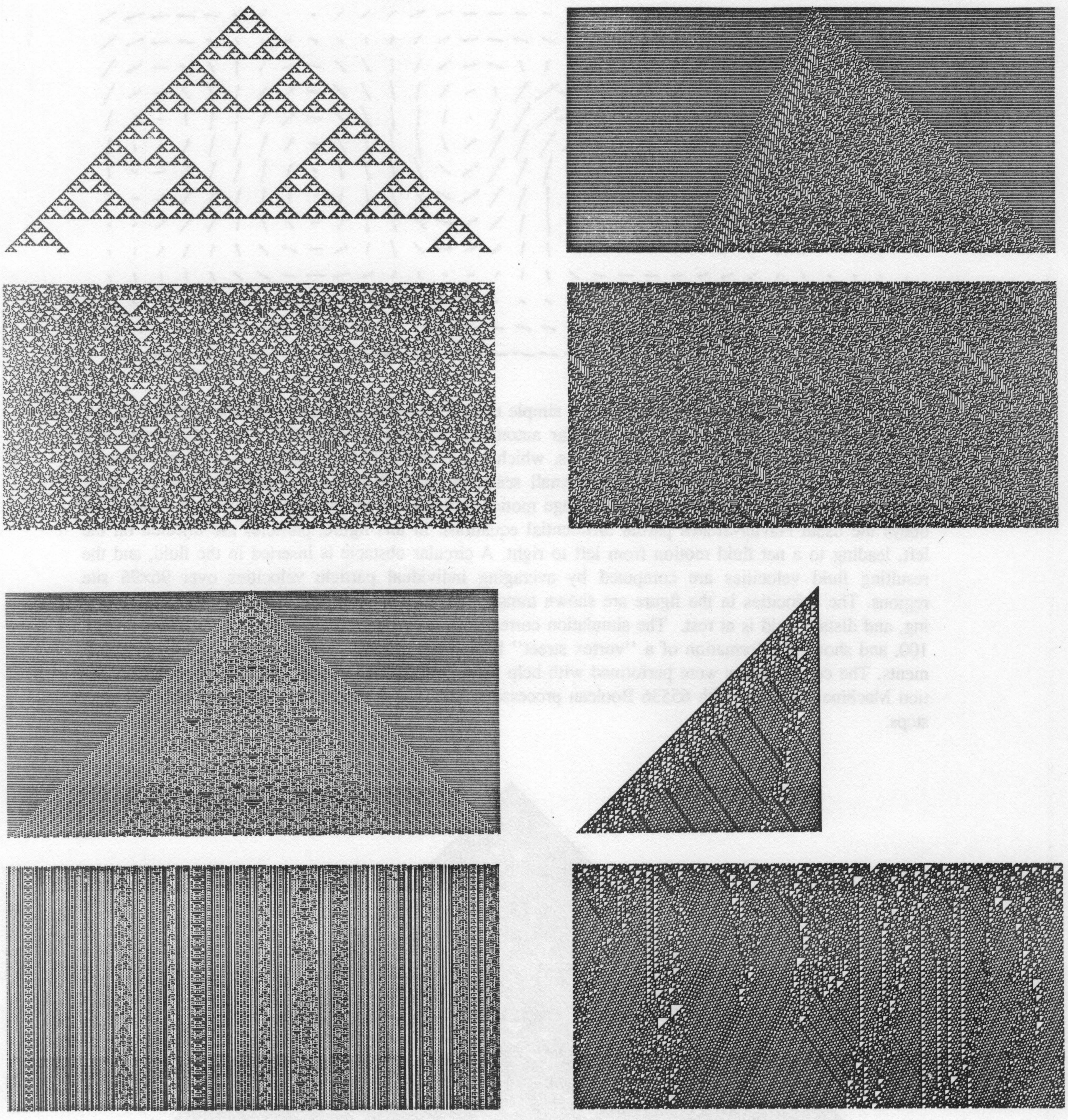

Figure 1: Examples of patterns generated by simple one-dimensional cellular automata. The cellular automaton consists of a row of about 600 sites, whose values evolve with time down the page according to simple logical rules. The value 0 or 1 of each site (represented by white or black) is determined from its own value, and the values of its two nearest neighbours on the step before. Patterns generated by four different rules are shown. In each case, the pattern obtained with an initial state containing a single nonzero site is shown above, and a pattern generated with a random initial state is shown below. (In the notation of ref. [3], the rules are numbers 18, 45, 73 and 110.) Despite the simplicity of these cellular automata, the patterns generated show considerable complexity.

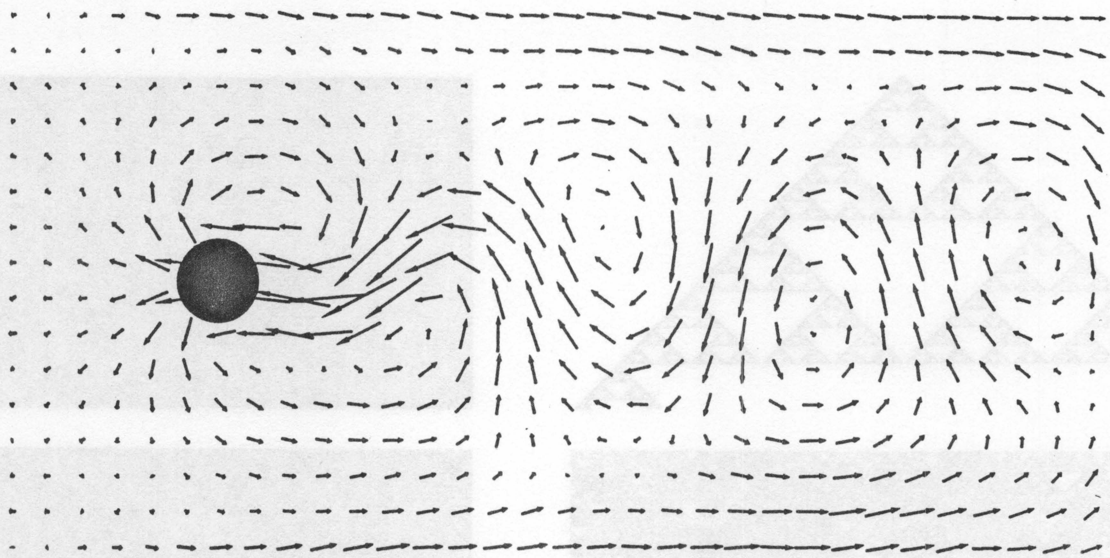

Figure 2: Fluid flow pattern obtained from a simple two-dimensional cellular automaton, simulated on a Connection Machine computer. The cellular automaton consists of 4096×2048 site hexagonal grid. Each site carries up to six discrete particles, which move and collide according to a simple discrete idealization of molecular dynamics. On a small scale, the particle motions appear random. But on a large scale, there is evidence that their average motion corresponds to that expected from a fluid which obeys the usual Navier-Stokes partial differential equations. In this figure, particles are injected on the left, leading to a net fluid motion from left to right. A circular obstacle is inserted in the fluid, and the resulting fluid velocities are computed by averaging individual particle velocities over 96×96 site regions. The velocities in the figure are shown transformed to the frame in which the obstacle is moving, and distant fluid is at rest. The simulation corresponds to a dimensionless Reynolds number around 100, and shows the formation of a "vortex street" behind the cylinder, as observed in physical experiments. The computations were performed with help from Bruce Nemnich and Jim Salem, on a Connection Machine computer with 65536 Boolean processors. The results shown were obtained after 10^5 time steps.

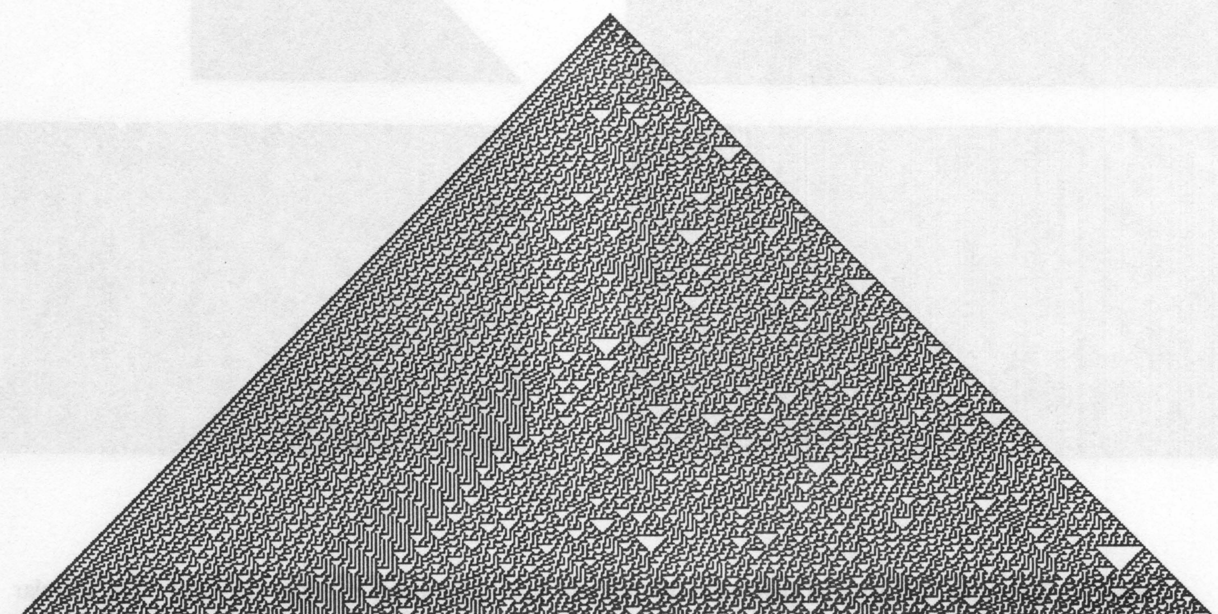

Figure 3: Pattern generated by a one-dimensional cellular automaton with two possible values at each site, and rule $a'_i = (a_{i-1} + a_i + a_{i+1} + a_i a_{i+1})$, starting from a single nonzero site. Despite the simplicity of its specification, many aspects of the pattern seem random. For example, the center column of site values passes all standard statistical tests of randomness. This cellular automaton illustrates the rather general phenomenon that simple processes can lead to complexity that is so great that many aspects of it seem random.

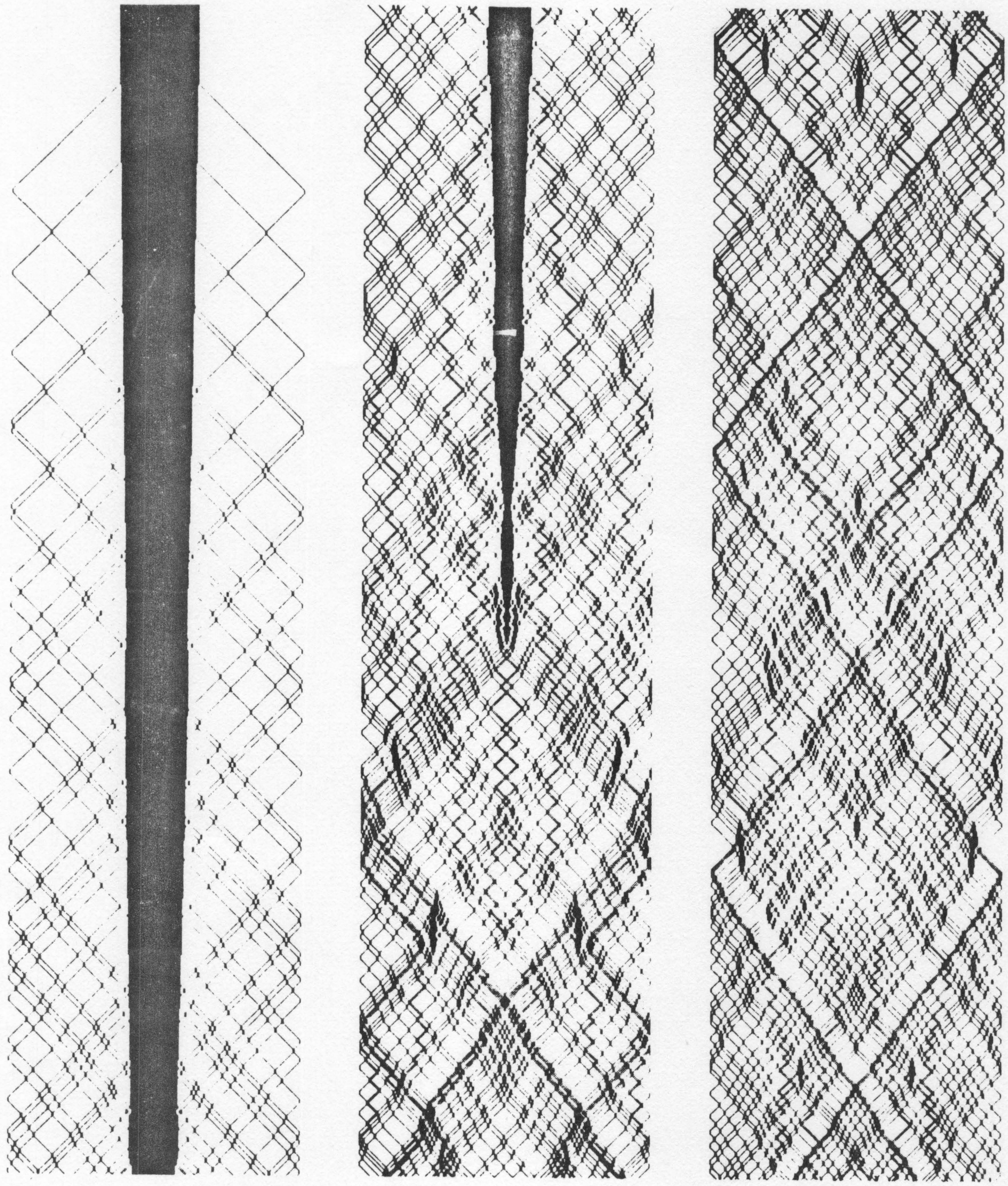

Figure 4: Pattern generated by a simple cellular automaton rule intended to mimic one-dimensional diffusion. Starting from a simple initial state, the reversible cellular automaton rule yields states that seem progressively more random. Such behaviour corresponds to that expected from the Second Law of thermodynamics, and can form the basis for simple discrete cellular automata to show macroscopic average behaviour which mimics continuum phenomena.