Journal of Statistical Physics, Vol. 58, Nos. 3/4, 1990

On Demonstrating Cooperative Phenomena

D. C. Rapaport[1]

Received July 14, 1989

The benefits of computer simulation can be enhanced by including animated displays showing the behavior of the systems under study together with the capability for responding immediately to parameter changes and other modifications requested by the observer. A modern computer graphics interface is used to facilitate user interactivity at a highly intuitive level. While hardware capability still limits the nature of problems that can be treated in this manner, the approach provides a much richer simulation environment than previously available. A number of examples of relevance to statistical physics are discussed, the majority of which are associated with cooperative behavior in interacting many-body systems.

KEY WORDS: Computer simulation; cooperative phenomena; interactive computing; computer graphics.

1. INTRODUCTION

After an excellent working relationship spanning some four decades, the computer has become an indispensable tool in most branches of physical science. To the experimental scientist the computer provides the means for controlling and monitoring the apparatus (be it a box on the laboratory bench or an accelerator several miles in length) as well as being responsible for data analysis and presentation of results. Much of modern experimentation would be totally impossible without the aid of the computer, a far cry from the string and sealing wax days of yore.

For theoretical science the impact of the computer has perhaps been even more profound. While there is no question that the computer has proved indispensable for its powerful calculational capability, not to

[1] Department of Physics, Bar-Ilan University, Ramat-Gan, Israel.

Dedicated to Cyril Domb—teacher and colleague—on his 70th birthday.

0022-4715/90/0200-0775$06.00/0

mention its role as a symbol manipulator to aid in dealing with algebraic aspects of theory, a particularly important role now filled by the computer is as a simulator of physical phenomena. Whereas theoretical development often relies heavily on judicious approximation, simulation is often able to provide the means for investigating a problem in a form free of simplifying approximations, the possibly deleterious effects of which are difficult to assess.

One example of the contribution of the computer to a field familiar to the reader involves critical phenomena.[1] While the study of phase transitions extends back over a century, it was only with the solution of the two-dimensional Ising model that it became clear that statistical mechanical theory could actually produce the singular behavior associated with a second-order transition. While there have been further developments based on analytic methods, a great deal of the subsequent progress, including the provision of empirical evidence on which the modern theory of critical phenomena is based, has depended on considerable computational effort. Two major routes have been followed. One attempted to probe the singularities associated with the transition directly by means of power series expansions; diagrams were introduced to sort out the contributions to the series[2] and though the processing was initially done manually, it was soon realized that the computer was more than capable of dealing with this problem.[3,4] The other route involved the Monte Carlo sampling of phase space for the system under study.[5] While generally (though not always) less precise than series methods for studying critical properties, the method has proved extremely valuable in exploring a very broad range of lattice models. As normally carried out, the Monte Carlo method provides a certain degree of continuity between the successive states generated, and is thus justifiably regarded as a simulation of the system, rather than a mere evaluation of a phase-space integral—which indeed it also is. Since most lattice systems (such as those based on the Ising model) do not possess any inherent dynamics, the passage through phase space made possible by Monte Carlo is often regarded as the "dynamics" of these systems.

Another example is provided by the liquid state. Approximate theories abound, based on methods including self-consistent approximations and perturbation expansions.[6,7] The Monte Carlo approach has proved extremely useful in studying the equilibrium properties of liquids in an analogous manner to lattice systems.[5,8] But with liquids there is an alternative approach, namely molecular dynamics.[8,9] The molecular dynamics method is also a means for sampling phase space, but the sampling is done by allowing the system to follow the path determined by its intrinsic dynamics, rather than by some artificially imposed scheme. By simply solving the equations of motion of the constituent molecules of the fluid, it

is possible to observe not only its equilibrium properties, but also a variety of nonequilibrium effects, such as transport, relaxation, and the formation of dissipative flow structures.

A reasonable assessment of the situation in both the examples cited is that a substantial proportion of the overall progress is due to the help provided by the computer, and of this progress the contribution of the simulational approach cannot be ignored. Simulation has often been described as an alternative way of doing science, as distinguished from the more established theoretical and experimental approaches. But in fact simulation is really just a new element in the theorist's toolkit—now one merely computes the properties of the model directly rather than becoming involved in approximations while working with the mathematical formalism. The methodology of simulation shares with experiment many of the principles of design and control, while the analysis of the results also follows similar well-established procedures.

In addition to its exploratory function in research, simulation can also be used as a means of demonstration in an educational framework. Thus, to continue with the examples already introduced, a feeling for the behavior of spin models and liquids can be conveyed by actually demonstrating the simulations, in much the same way as an experiment might be carried out in front of an audience. In the past, when the power of computers that could be brought into a lecture hall was less than adequate for conducting a simulation in "real time," the alternative of prerecording the results on film or tape might have been employed. This approach, however, permanently freezes the simulation and, unlike the experimental analogy, eliminates the capability for exploration in response to spectator enquiry. The obvious alternative is interactive simulation, where the computations are carried out on demand, and the response to requests for some change in the conditions of the simulation (within limits) can be immediate. The increased flexibility allowed by this approach is a direct consequence of the availability of greater computing power in smaller, cheaper packages than before.

Another important aspect that enhances the role of simulation as a demonstrational device is the availability of animated graphics. The scientist may tend to view graphics in its most basic role, as a means for conveniently summarizing tabular data. But in fact graphics goes far beyond this primitive function and allows the direct visualization of the phenomena being explored. Published examples of such uses of graphics abound; examples related to fluids and lattice models include pictures of atomic trajectories in liquids[10] and cluster development in percolation.[11] What the printed medium lacks, however, is the ability to animate. The human eye can extract a great deal of additional information from a

dynamical display in which temporal continuity is maintained, far more than is contained in the corresponding discrete sequence of snapshots.

The only shortcoming in the use of interactive simulation is the comparatively limited power of computers that can be reasonably—in an economic sense—devoted to such a task. A large-scale simulation can often occupy a late-model supercomputer for a good many hours. Given that the computers available for demonstrations are several orders of magnitude less powerful, and that the entire duration of the simulations must be compressed into the attention span of the audience, the scope of problems that are candidates for such treatment is constrained. Nevertheless, the problems that can already be addressed are both interesting and significant, with expected advances in computer performance available at reasonable cost, promising significant improvements in future.

In this paper, I begin by outlining some of the scientific and technical issues involved in implementing such demonstrations. A series of examples then follows, taken mainly from problems encountered in statistical physics. These problems have been extensively studied by means of simulation, some over a period of several decades. The interactivity and animation incorporated in the present approach introduce an extra dimension that can provide, even to the seasoned simulator and in spite of the limited scales of the systems modeled, a greatly enhanced picture of what is happening. Certainly for the beginner, as a means of conveying an appreciation of what a model system can do, the approach has few rivals; it is unfortunate that the printed medium is unable to convey the full flavor of the visual experience. Finally, the particular computer used in this work, the Commodore Amiga,[12] is discussed in an appendix.

2. SIMULATION

The simulational approach can have a variety of different goals. In some instances simulation might be used to test whether a model system can help in understanding an experiment; for example, there have been many studies of liquid structure[6,7,9] in order to understand neutron scattering data (two frequently studied liquids being water and argon). Similarly, again using liquids as an example, simulation can be used to test the range of validity of approximate theories.[6] In the case of liquid studies the models used are attributed a certain degree of realism, while in the case of a growth process such as diffusion-limited aggregation (DLA), the model is highly schematic, but still appears to contain enough relevant detail to produce reasonably familiar growth patterns.[13] Yet other model systems, such as cellular automata,[14] may be even more remote from the real world which inspired their creation, yet the hope is that certain

meaningful patterns of behavior may emerge which could shed light on the original problem, even though the link may be rather tenuous. Examples of each of the above simulations are discussed later.

The techniques of simulation are manifold. Whereas fluids can be simulated by means of both Monte Carlo and molecular dynamics, most (though not all) spin systems have no inherent dynamics and can only be studied with the former technique. These are instances where the simulation method can be regarded as distinct from the model. In other situations the simulation method and the model are synonymous; examples include DLA (one could of course employ a more complex method, such as molecular dynamics, to grow clusters) and cellular automata, both of which embody the method of simulation in their definitions. The computational techniques used in the simulations are reasonably familiar; descriptions are spread across a variety of sources, including many of the references cited, and for this reason only a brief outline of the methodology is included here.

3. INTERACTIVITY AND ANIMATION

The requirement for displaying the evolution of the system and the capability for making changes in the course of the computation are both novel demands not encountered in simulation as usually practiced. The normal state of affairs is that a computer run is submitted with established parameter settings and the outcome is returned as a set of numerical results that can be tabulated, graphed, or otherwise processed, but only after the simulation is complete. Of course, intermediate results could be used to decide how to alter the parameters for the next stage of the simulation, but the scope for this kind of interaction is often limited.

The situation here is very different. The user is free to alter the parameters of the problem at will, as well as making other kinds of changes that a simulation might be designed to accommodate. The structure of the simulation algorithm must allow for such eventualities, and this indeed tends to complicate the development of this kind of software. The nature of the output also differs considerably from the more traditional forms of simulation, to the extent that numerical output might be entirely eschewed in favor of displays of the actual state of the system. Thus, for example, instead of supplying the spin–spin correlation function for a set of classical vector spins, the actual rotational motion of the spins is shown; if any obvious correlations are present, it is the task of the observer to discover their existence (a not especially demanding job). Of course, it would not be difficult to provide such information, since the data on which it is based are

obviously available, but the whole idea of the present work is to demonstrate the unadorned phenomena as they actually occur.

The interaction with the system is also carried out in a manner that is both straightforward and intuitive. Typing from the keyboard is almost entirely avoided, and the dialogue employs a mouse (a nonbiological device which, when pushed around on a flat surface, informs the computer of its position, and also signals the state of its one or more pushbuttons whenever changes occur) linked to a pointer on the screen. Under mouse control it is possible to select from a choice of actions listed on the screen merely by pointing at them; these actions can appear in the form of icons on permanent display that are touched with the pointer, or as sets of menu items available for selection that can be summoned on request. There are a great many ways of entering data. One particularly appealing method is the use of icons in the form of sliding "potentiometers," the knobs of which can be dragged to any desired position using the mouse; these can be regarded as analog input devices. Another possibility is the selection of a value from a predetermined set; values could of course be entered from the keyboard if this is deemed most appropriate.

4. DEMONSTRATIONS

In this section I describe a selection of the simulations that have been implemented. The focus is on simulations that are directly relevant to statistical physics, but examples of problems that only marginally fall within the field—the criterion being the subject matter covered by this journal—have also been included to broaden the scope a little. (A broad range of simulations that address more basic problems in physics have also been developed, but these lie outside the theme of this paper; a description of this other work has been published elsewhere.[36]) Each of the simulations is described from several points of view: the problem that is modeled and (briefly) the techniques used, what the user observes, and the capabilities for interacting with the system.

4.1. Ising Model

The Ising model is a system that has been exhaustively investigated by means of Monte Carlo simulation.[5] The version of the system considered here is the simplest two-dimensional case—the exact zero-field solution in the limit of infinite lattice size is well known. The system consists of a 4096-site square lattice with ferromagnetic interactions and an external field. Both the temperature and field are under user control; the values are changed using a pair of sliding "potentiometers" of the type described

earlier. System response to changes of value is immediate. The initial state of the simulation can be random or ordered, and a scrolled display of a graph showing the development of short- and long-range order (the energy and magnetization, respectively) can be requested. The simulation technique used is standard Metropolos-type Monte Carlo, with the lattice being swept in a systematic fashion; color coding is used to differentiate between up and down spins, and the display is refreshed following each complete pass through the lattice. The computation proceeds at a rate of approximately three passes per second; though this is over four orders of magnitude slower than the fastest implementations of the same calculation that run on supercomputers (respective hardware costs are in a similar ratio), it is the other features of this particular implementation that compensate for the lack of raw speed.

What the user sees depends on the way the temperature and field are manipulated. One of the possible scenarios entails gradually reducing the temperature from some initially high value with the applied field set to zero. Here it is possible to observe the behavior as the system moves through the critical region—cluster fluctuation and growth—as well as the relatively rapidly varying bulk properties that are shown on the graph. The initially small and wildly fluctuating clusters at high temperature gradually acquire longevity and a more regular form; cluster growth at low temperature is seen to be aimed at reaching the ground state as quickly as possible. The mechanisms by which clusters change their size and shape can be studied simply by observing the typical movements of the cluster boundaries. The ground state may be all up spins or all down; the periodic boundaries also permit the system to lock itself into a two-phase state, with a linear interface that cannot be eliminated, except by reheating (or by applying a field). Another kind of behavior that can be studied is cluster nucleation in an external field. The system is cooled to low temperature with the field applied in one particular direction; the field is then reversed and the initially small fluctuating clusters of reversed spins that appear can be seen to grow, eventually leading to a complete state reversal.

4.2. Vector Spin Model

This simulation addresses an alternative kind of spin model studied by a different technique. The system is based on the planar classical Heisenberg model[5] in zero field with interactions that can individually be set to favor parallel or antiparallel alignment. Each spin is a freely rotating entity (a rotor) whose center of mass is fixed at a lattice site; the lattice is a rectangle containing 48 spins with boundaries that are periodic. The behavior of the system is followed by solving the rotational equations of motion,[15]

in which the torques acting on the rotors depend on the relative alignment of each rotor with respect to its neighbors; the response depends on the moment of inertia of the rotor. The numerical integration of these coupled differential equations is carried out using a simple leapfrog scheme.[16] The system is studied in the microcanonical ensemble; the mean temperature can be controlled by injecting or removing "bursts" of kinetic energy (just a simple rescaling of angular velocities). The signs of the interactions (that dictate the preferred alignment) can be controlled by requesting the display of a superimposed color-coded grid and then using the mouse to select the bonds corresponding to the interactions to be altered.

At high temperature what is seen amounts to free rotation, and the animation techniques employed ensure visually smooth spin motion. As the temperature is lowered, there is a growing tendency for adjacent spins to align. Spin waves can be observed traversing the system at sufficiently low temperatures, and eventually the system locks into an ordered state (the system is finite, so there can be no objections to behavior of this kind); the entire system is usually found to precess slowly—a consequence of the residual net angular momentum from the initial state. If the system is now halted and some of the interactions switched from ferro- to antiferromagnetic, the subsequent behavior when the simulation is restarted depends strongly on which bonds were changed. Certain kinds of systematic patterns of the two interaction signs can lead, for example, to antiferromagnetic or layered states, but if the changes are made at random, the system will be observed to exhibit frustration,[17] and the interested viewer can watch to see how the system copes with the situation.

4.3. Soft-Disk Fluid

A two-dimensional fluid of disks is about the simplest model system for reproducing many of the characteristic features of liquids and gases. An enormous body of knowledge has accrued as a result of both molecular dynamics and Monte Carlo studies of simple as well as more complex fluid models.[6,7,9] Despite its rudimentary appearance, the soft-disk fluid is a useful vehicle for a broad range of problems ranging from melting transitions to transport phenomena. While full-scale simulations deal with systems ranging from a few hundred particles (be they disks, atoms, or molecules) to 200,000,[18] the constraints of real-time performance and the limited computational power of the available hardware confine the simulations here to a size of about 100 (the actual upper limit has been set to 150).

The observer is presented with a container containing disks of two colors. The two kinds of disks are identical—the colors are provided as an

aid for monitoring mutual diffusion of the two species. The size and shape of the container can be altered by dragging one of its corners; since the density at the start of the run is fixed, the initial container size governs the number of disks present. There are gauges showing temperature and pressure as computed from the mean kinetic energy and virial respectively,[7] together with a set of "button" icons which provide facilities for heating and cooling the system as well as for activating a display of trajectories.

The disks interact via a fairly strongly repulsive potential, which, though it allows a certain amount of overlap whenever a fast collision occurs, quickly forces the disks apart, although this process can be hindered by a rapid sequence of multiple collision events (more likely to occur at high density). Solution of the equations of motion is by means of standard integration methods; here the leapfrog method is again employed. Periodic boundaries are used, so that disks leaving through one edge of the container are seen to reappear at the opposite edge. The simulation progresses at a rate that obviously depends on the number of disks present [a cell-list scheme (e.g., ref. 19) is used to ensure a linear dependence of the computation time on the number of disks]; provided the system is not too large, the animated motion of the disks appears relatively smooth, but for the largest systems the updata rate drops to about two time steps per second, which, though rather uneven, is on the borderline of what is tolerable.

By manipulating the temperature and container volume, the observer can study a variety of different aspects of fluid behavior: the processes of freezing and melting, clustering, the temperature and density dependence of diffusion, and the nature of the particle trajectories at high and low densities (a visually impressive result generated in the earliest[10] molecular dynamics simulations) are examples of the effects that can be explored.

4.4. Lattice Vibrations

This particular simulation demonstrates the behavior of a set of masses and harmonic springs[20] linked together in the form of a triangular lattice. A total of 72 masses are used, and the lattice boundaries are periodic. The coupled equations of motion of the masses are solved numerically as before. The user is given control of the temperature of the system (defined to be the mean kinetic energy of the masses), and is also allowed to remove as many of the springs as desired—spring removal is accomplished with the system at rest merely by using the mouse to point at the spring in question.

The initial motion of the masses corresponds to small vibrations at

low temperature, but as the system is heated it is possible to see short-lived displacement waves crossing the lattice. At higher temperatures the motion becomes quite violent. When springs are eliminated other kinds of behavior are noted; in particular, if an island of connected masses is created that is only linked to the rest of the system by a few well-separated springs, it is possible to see the entire island pulsating in a variety of collective "breathing" modes. Such behavior is yet another example of a phenomenon that is best appreciated through direct observation.

4.5. Diffusion-Limited Aggregation (DLA)

DLA is the simplest model for studying growth processes.[13] This particular kind of mechanism leads to the formation of highly ramified aggregates reminiscent of coral and soot. The growth takes place on a two-dimensional lattice, and involves a fixed seed particle around which the cluster will grow, together with a succession of diffusing particles which, upon collision with the cluster, adhere to it. Diffusion is represented by a simple random walk between adjacent sites. Pictures of the patterns that emerge from studies of this type abound in the literature.[13,21]

In this simulation the observer sees the growth process as it actually occurs. Each diffusing particle moves randomly around the lattice at a rate of approximately 15,000 steps/second; this is too fast for either eye or display to follow, so the displayed particle position is only updated once every 500 steps. Differently colored shells are used to delineate the temporal development of the cluster. The number of times a site must be visited before a particle can adhere to the cluster can be increased from the default value of unity, and the color scheme used to follow the growth can be freely modified (for example, the cluster interior can be set to the background color, so that only the peripheral growth region is visible). Despite the fact that the square lattice used is only 180 sites on an edge, a good impression of the variety of cluster shapes possible can be obtained before the viewer's patience expires.

4.6. Percolation

Percolation[22] is another example of a problem (a nonanimated one) that deals with clusters, although here it is not a case of a growth process, but one of connectivity between lattice sites (only the site problem is treated) each of which decides independently whether it is capable of joining the cluster—in the usual language, the criterion is whether the site is occupied or vacant. The observer is shown the entire lattice, with the sites painted in different colors that distinguish occupied from empty. While the

eye can at least partly determine where connected clusters of occupied sites are to be found, the largest four clusters are shown in different colors. The so-called occupation probability can be controlled by setting to the desired value; the dramatic growth in the size of the largest cluster as the critical probability is passed can be clearly seen.

There are a number of ways of approaching the percolation problem; the fact that the goal of these simulations is actually to see the clusters prescribes the means of going about the task. For a given simulation run, instead of each lattice site being occupied or vacant, the sites are assigned values uniformly distributed between zero and unity; then, when an occupation probability value is set, all sites whose assigned values lie below this setting are deemed to be occupied. Thus, it is possible to scan over a range of probabilities and observe how the clusters develop. The entire set of assigned values can be changed at will. The lattice size is a square of edge 160, and periodic boundaries are used to avoid hindering development of larger clusters. The algorithm used to determine which sites belong to which clusters is a well-known one.[4,23]

4.7. Reptating Chains

Reptation is a mechanism proposed for the motion of linear polymers in a melt or concentrated solution. To what degree it fully characterizes the dynamics is still a subject for exploration.[24] The idea of reptation has also been adopted as a means for simulating polymer chains on the lattice; while the process is not entirely ergodic, it does produce results for configurational properties that agree with other methods.

The present simulation is included as much by way of amusement as for any other reason. Four long (320-link) or 16 shorter (100-link) self- and mutually-avoiding chains, initially in the form of square spirals, are placed on a lattice with box boundaries. The chains are allowed to undergo reptational motion: One end of each of the chains—the head—takes a randomly selected step to an unoccupied site, and the rest of the chain follows; if it cannot find such a site, then the other end of the chain takes over the role of head. It is quite an enlightening experience to observe the behavior of the chains; even though they have no capability for looking ahead and no memory, they eventually succeed in escaping from their initial configurations and move about the lattice, albeit in a very inefficient manner. Some of the ways in which the chains can become entangled, both with themselves and with one another, should serve as a warning as to the pitfalls of this kind of simulation mechanism.

4.8. Cellular Automata

Cellular automata represent a very broad category of models used in the study of a variety of phenomena, including cell growth, pattern formation, and hydrodynamics.[14] The class of automata dealt with here are two dimensional and based on the simplest transition laws, the so-called totalistic rules, in which only the sum of the neighbor states determines the evolution of a cell. An extremely rich variety of patterns can be generated using rules of this class, and a certain amount of systematic exploration has been carried out. The well-known game of Life[25] is a particular instance of this automata class. While the present software implementation cannot complete in terms of performance with special-purpose hardware,[26] it does of course avoid the additional expence and still gives an adequate impression of the properties of the model.

The simulation employs a 160-squared grid, and updates of the entire system are made at a rate of approximately one every 2 secs. The two states of each cell—dead or alive— are shown in different colors, with an option for additional colors to show cells that have died in recent previous generations and have not yet been resurrected (a kind of short-term historical record). Since this simulation is designed to allow exploration of an entire range of automata problems, menus provide the user with the ability to choose the features of the system: whether the neighborhood of a cell used to determine its next state involves either four or eight adjacent cells; whether the state of the cell itself is treated together with or separately from its neighbors in determining the next state; the actual rule—depending on the model,[27] there can be up to some quarter million totalistic rules; and whether the initial state is a single live cell, a small random cluster of live cells whose size can be specified, or a random arrangement of live cells covering the entire lattice with an adjustable overall concentration. The observer is then invited to sit back and enjoy the surprising effects that develop on the screen and wonder about the exceedingly simple mathematical mechanism responsible for such elaborate patterns; the broad range of behavioral patterns cannot be concisely summarized.

4.9. Traveling Salesman

The traveling salesman problem[28] is a paradigm for the kinds of combinatorial optimization problem faced (for example) in planning airline routes, telecommunication networks, and printed circuit board layouts. The problem is to find the shortest closed path that makes a single visit to every one of a specified set of cities; as usually formulated, the cities are randomly located in a rectangular planar region. The general problem does

not admit an efficient algorithmic solution (the problem is in the *NP*-complete class[29], but a variety of heuristic approaches are available that can produce near-optimal solutions quite effectively.

One comparatively recent approach is known as simulated annealing,[28] and is a technique inspired by the Monte Carlo approach as used, for example, in the Ising model.[5] In its simplest form the algorithm for solving the problem[30] is as follows: Start with a randomly selected route that visits all the cities. Each step of the algorithm requires that a pair of cities be picked at random and a trial change in the path made so that the sequence of cities between the chosen pair is visited in reversed order. If the new total path length is shorter, it is accepted in place of the old, but if the path length is greater, it may still be accepted, conditional on the successful outcome of a test that is formally identical to one used in the Ising model simulation (with path length used in place of energy). There is a control parameter in the simulation whose role corresponds to that of temperature in the Monte Carlo study of physical systems; regulation of this parameter determines whether the path changes favor major modifications to the route or only minor adjustments. The scheme for adjusting the control parameter to attain best overall performance is known as an annealing schedule.

The user is allowed to request as many as several hundred cities that are randomly distributed on the screen. Beginning with an excessively long initial random path, each path update is shown, and the gradual convergence to the minimal path can be followed. It is up to the user to decide what values of temperature to use and when to change them. The time required for convergence depends on the number of cities—from a few seconds for a very small number of cities to about 10 minutes for a hundred. To check that the shortest path is reproducible, the temperature can be raised to generate another long random path and another optimization carried out. If there is near-degeneracy, in the sense that several distinct paths are close to minimal length, then there is of course no way of controlling which will be selected.

4.10. Logistic Map

Chaos[31] has become a subject of major interest to the statistical mechanical community in the past decade. While chaotic behavior can be detected in an enormous variety of dynamical systems, it is most readily appreciated when observed in a very simple context. A simple nonlinear algebraic recurrence known as the logistic map[32] provides such a framework. While static pictures of the bifurcations that correspond to period doubling are familiar to all, the goal of this particular implementa-

tion of the calculation is to allow exploration of the entire parameter range of the problem in an interactive way; the extra dimension provided by the ability to observe how the iterations actually converge—if at all—is an important part of comprehending the problem. For example, there is no effective way that static pictures can convey a feeling for intermittency (where periodicity and chaos alternate), nor for the abruptness of the transition between chaos and periodic behavior (such as when cycles of three, five, and so on, suddenly materialize).

The initial display shows a blank region upon which the results of the computations are displayed in cumulative fashion. The parameter setting is chosen by moving a cursor across the screen. When a value is selected the iteration process begins immediately, and continues until the parameter setting is changed. The initial value for the iterated variable is picked at random, and over much of the parameter range convergence to the limiting value is of course extremely rapid. The points shown on the display represent the 30 latest values produced by the computation. The full parameter range, subject only to the limited resolution of the display, can be explored in this way. In order to examine a particular region of parameter space in greater detail, it is possible to zoom in on a selected subrange simply by entering the desired parameter limits. In addition, to aid in following the history of each series of iterations, the observer can request an additional display in which the horizontal axis is the iteration number rather than the parameter setting; the generated values scroll smoothly across the display, making it particularly easy, for example, to observe the nature of intermittency. Yet another aid in analyzing the behavior is spectral analysis; the observer can request a power spectrum of the iteration sequence that is recomputed at regular intervals—peaks in the spectrum (in the appropriate parameter ranges) reveal periodicities in the apparent chaos that might otherwise go undetected.

5. THE FUTURE

The studies described in this paper are in many instances scaled-down versions of much larger problems that have been, and in some cases still are, subjects for serious research activity. The kinds of simulation that can now be carried out in an interactive manner on a desktop computer are typically of a size that, not so many years ago, would have been perfectly adequate for generating results of publishable quality. If one naively extrapolates the progress of computer technology forward a decade, one might expect to see today's state-of-the-art simulations carried out on the desktop computers of the day. Whether this will actually come to pass is

another matter; but certainly one can expect to see considerable performance enhancement in this category of affordable computer. As for the benefits of visualization and interactivity, they speak for themselves.

APPENDIX. METHODOLOGY AND HARDWARE

The computer used in this work is the Commodore Amiga.[12] There are several models in existence at the time of writing (including the 500, 2000, and older 1000), but for most purposes these are functionally equivalent. (Enhanced performance has become available recently, but the cost increment is substantial.) A feeling for the cost of the hardware required for these demonstrations should follow from the fact that the price of a fully adequate system (including monitor) is below $1000. The processor used in the machine is the Motorola 68000 running at 7.2 MHz. While a number of other microcomputers also use this processor chip, what gives the Amiga its unique capability when it comes to animated graphics is an additional hardware element (a "bit blitter") whose sole purpose is the rapid moving and combining of entire blocks of data in a manner that proves extremely helpful for graphic display creation and manipulation.[33] Further technical information can be found in the literature.

All the simulations were written in the "C" programming language.[34] While Fortran remains the dominant language for scientific work, the use of C is more convenient on the Amiga because it meshes nicely with the graphic and other functions that are part of the system software. Indeed, the C language has a richer (though slightly more terse) syntax than standard Fortran 77, and is therefore easier to work with. Since the basic version of the machine does not include floating-point hardware, the simulations that rely on floating-point calculations (in particular, the dynamical studies) must make use of a software emulation facility, which of course slows down the calculations; in some instances floating-point computations were replaced with scaled integer arithmetic to improve performance.

A broad selection of graphics utility functions are provided for the machine.[35] On occasion, there may be more than one way of achieving a specific visual effect, especially in the realm of animation, and a certain amount of experimentation is required to determine which is preferable. Animation generally tends to be rather lavish in terms of memory requirements, so the fact that the typical Amiga tends to have no more than one megabyte of memory means that care is needed in memory utilization.

The interactivity is also made possible by the system software. For a computer to be able to deal with external events, such as a mouse move-

ment or icon selection, the program must be designed to provide prompt response in order that the user receive the impression of immediate feedback. This places demands on the simulation programs that are not present in software designed for the more familiar noninteractive batch production environment. A feature common to all the simulations is that the central part of each program is the one dealing with user interaction; it is only when user requests are not being serviced (which is in fact the majority of the time, but here the issue is one of priorities) that the simulation computations are performed. The system software[35] conceals many of the details associated with the interactivity from the application developer, and provided a certain systematic procedure is adopted, it is not too difficult to produce a simulation that allows the user a variety of choices at any stage. The problem is an organizational one common to all interactive computer applications, namely to prevent the user from communicating with the system in a nonsensical way; there is a need to ensure that the user is kept aware of exactly what options are available at each instant in order to avoid any misunderstanding.

One issue has been avoided, namely compatibility. The machine on which this work was carried out, though moderately popular, is not the most widespread in the scientific community; however, the more popular machines currently available in this price range do not have comparable graphic capabilities. It therefore appears necessary to have access to this particular machine to carry out demonstrations of this kind. In due course one can expect to see enhanced graphics performance in other affordable computers, and then the problem of software compatibility is certain to arise. Even though there exist fairly standardized programming languages, the situation with graphics is far from uniform. This means that in order to transport the software to other computers it will be necessary to convert the (often substantial) display and user-interface portions of the programs to the requisite forms.

REFERENCES

1. C. Domb and M. S. Green, eds., *Phase Transitions and Critical Phenomena*, Vols. 1–6; C. Domb and J. L. Lebowitz, eds. *Phase Transitions and Critical Phenomena*, Vols. 7– (Academic Press, New York, 1972–).
2. C. Domb, in *Phase Transitions and Critical Phenomena*, Vol. 3, C. Domb and M. S. Green, eds. (Academic Press, New York, 1974), p. 1.
3. J. L. Martin, in *Phase Transitions and Critical Phenomena*, Vol. 3, C. Domb and M. S. Green, eds. (Academic Press, New York, 1974), p. 97.
4. D. C. Rapaport, *Computer Phys. Rep.* **5**:265 (1987).
5. K. Binder, ed., *Applications of the Monte Carlo Method in Statistical Physics* (Springer-Verlag, Berlin, 1984).

6. J. A. Barker and D. Henderson, *Rev. Mod. Phys.* **48**:587 (1976).
7. J. P. Hansen and I. R. McDonald, *Theory of Simple Liquids* (Academic Press, New York, 1976).
8. M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids* (Oxford University Press, Oxford, 1987).
9. G. Ciccotti and W. G. Hoover, eds., *Molecular Dynamics Simulation of Statistical Mechanical Systems*, (North-Holland, Amsterdam, 1986).
10. B. J. Alder and T. E. Wainwright, *J. Chem. Phys.* **31**:459 (1959).
11. C. Domb, E. Stoll, and T. Schneider, *Contemp. Phys.* **21**:577 (1980).
12. T. Thompson, *Byte* (10) **11**:231 (1986); C. Heath, *Byte* (4) **13**:137 (1988).
13. L. Sander, *Nature* **322**:789 (1986).
14. S. Wolfram, *Theory and Applications of Cellular Automata* (World Scientific, Singapore, 1986).
15. H. Goldstein, *Classical Mechanics* (Addison-Wesley, Reading, Massachusetts, 1950).
16. H. J. C. Berendsen and W. F. van Gunsteren, in *Molecular Dynamics Simulation of Statistical Mechanical Systems*, G. Ciccotti and W. G. Hoover, eds. (North-Holland, Amsterdam, 1986), p. 43.
17. K. Binder and A. P. Young, *Rev. Mod. Phys.* **58**:801 (1986).
18. D. C. Rapaport, *Phys. Rev. A* **36**:3288 (1987), and to be published.
19. D. C. Rapaport, *Computer Phys. Rep.* **9**:1 (1988).
20. M. Born and K. Huang, *Dynamical Theory of Crystal Lattices* (Oxford University Press, Oxford, 1954).
21. H. E. Stanley and N. Ostrowsky, eds., *On Growth and Form* (Martinus Nijhoff, Dordrecht, 1986).
22. D. Stauffer, *Phys. Rep.* **54**:1 (1979).
23. J. Hoshen and R. Kopelman, *Phys. Rev. B* **14**:3438 (1976).
24. K. Kremer, G. S. Grest, and I. Carmesin, *Phys. Rev. Lett.* **61**:566 (1988).
25. E. R. Berlekamp, J. H. Conway, and A. K. Guy, *Winning Ways*, Vol. 2 (Academic Press, New York, 1982), p. 817.
26. T. Toffoli and N. Margolus, *Cellular Automata Machines* (MIT Press, Cambridge, Massachusetts, 1987).
27. N. H. Packard and S. Wolfram, *J. Stat. Phys.* **38**:901 (1985).
28. S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, *Science* **220**:671 (1983).
29. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms* (Addison-Wesley, Reading, Massachusetts, 1974), p. 364.
30. W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes* (Cambridge University Press, Cambridge, 1986), p. 326.
31. P. Cvitanovic, ed., *Universality in Chaos* (Adam Hilger, Bristol, England, 1984).
32. R. M. May, *Nature* **261**:459 (1976).
33. *Amiga Hardware Reference Manual* (Addison-Wesley, Reading, Massachusetts, 1986).
34. B. W. Kernighan and D. M. Ritchie, *The C Programming Language* (Prentice-Hall, Englewood Cliffs, New Jersey, 1978).
35. *Amiga ROM Kernel Reference Manual: Libraries and Devices* (Addison-Wesley, Reading, Massachusetts, 1986); *Amiga Intuition Reference Manual* (Addison-Wesley, Reading, Massachusetts, 1986).
36. D. C. Rapaport, *Computers in Physics* **5**:19 (1989).