Simulating digital logic with the Reversible Aggregation model of cluster growth

Raissa M. D'Souza
Department of Physics, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139

George E. Homsy
Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139

Norman H. Margolus
Center for Computational Science, Boston University, Boston, Massachusetts 02215
and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139
(September 12, 1999)

We are concerned with understanding the implicit computation occurring in a physical model of cluster growth, the Reversible Aggregation model. The RA model is a lattice gas model of reversible cluster growth in a closed two-dimensional system, which captures basic properties of physics such as determinism, locality, energy conservation, and exact microscopic reversibility. There are three species of particles in the RA model: gas, heat, and cluster. A diffusing gas particle may aggregate when contacting the boundary of a connected cluster. Latent heat is released during each aggregation event and is explicitly modeled by introducing a heat particle into a diffusing heat bath. Conversely a cluster particle at the boundary of the connected cluster may absorb a heat particle and evaporate, becoming a diffusing gas particle.

Allowing ourselves complete control over all the initial conditions of the model, we show that the RA model can simulate any logic circuit, and hence perform any computation. The mobile gas and heat particles are used as logic signals. The paths these particles take are the wires. Sequences of conditional aggregation events form the basis of the logic gates. We show how to embed a universal single use gate into the dynamics of the model, then show how to construct a reusable universal gate, showing the system is capable of space-efficient computation. We show how to build arbitrary logic circuits by interconnecting gates. This requires steering and routing the signals, delaying them, and letting them cross. Finally we briefly discuss the relationship of computation in the RA model to computation in real physical systems.

I. OVERVIEW

We examine the computational capabilities of a physical model of cluster growth, the Reversible Aggregation (RA) model [1], which captures basic properties of physics such as determinism, locality, energy conservation, and exact microscopic reversibility. The RA model is a lattice gas model of reversible cluster growth in a closed two-dimensional system. It was introduced as a microscopically reversible physical model for studying the thermodynamics of cluster growth and pattern formation. By microscopically reversible we mean that from any state in the system we can recover the previous state exactly. There are three species of particles in the RA model: gas, heat, and cluster. A diffusing gas particle may aggregate at the boundary of a connected cluster; if it lands next to a single nearest neighbor cluster particle, it may aggregate, becoming a cluster particle and enlarging the connected growth cluster. Latent heat is released during each aggregation event and is explicitly modeled by introducing a heat particle into a diffusing heat bath. Conversely if a heat particle contacts a singly connected cluster particle it may be absorbed and that cluster particle will "evaporate" from the connected cluster, becoming a gas particle. When started with a dilute gas and a single cluster seed particle the model exhibits an initial regime of rapid nonequilibrium growth followed by a slow quasistatic regime with a well defined temperature. In the first regime a connected cluster rapidly grows, in the second the connected cluster slowly anneals.

The present study, showing a construction for computational universality in the RA cellular automaton seems fitting for these proceedings of a workshop on "Constructive Cellular Automata". The mobile gas and heat particles are the logic signals used in the computation. The paths these particles take are the wires. We show how to steer and route the signals, how to delay them, and how to allow signals to cross each other. Aggregation events occur only at sites with one nearest neighbor which is already a cluster particle. By routing a control signal through a potential aggregation site we can conditionally create new potential aggregation sites. Such sequences of conditional aggregation events are the basis of our logic gates. We show how to embed into the dynamics of the RA model a universal single use gate, and then how to embed a reusable universal gate. We show how to interconnect gates and thus how to build arbitrary digital logic circuits, proving that the RA model is capable of space-efficient computation.

In this manuscript we take a microscopic perspective; that is we give ourselves control over all of the microscopic degrees of freedom, namely the initial positions of all the gas, cluster, and heat particles, and detailed control over the microscopic parameters controlling the pseudorandom motion of the gas and heat particles. With microscopic control and synchronous time evolution we can compute with this discrete lattice system. A more general issue is understanding computation in real physical systems where we have control only over macroscopic degrees of freedom and where we cannot depend on perfect synchronization. In the final section we will address issues of whether we have abstracted concepts from our microscopic dynamics which apply to computation in real physical systems.

II. THE REVERSIBLE AGGREGATION MODEL

The RA model is a reversible, deterministic model of cluster growth in a closed two dimensional lattice system. It extends the canonical Diffusion Limited Aggregation (DLA) model of cluster growth on a lattice [2]. The DLA model is an irreversible, deterministic model with two particle species: gas and cluster. The gas particles follow a pseudorandom walk along the lattice sites, resulting in diffusive behavior at the scale of several lattice sites. If a diffusing gas particle contacts a stationary cluster member, it aggregates, itself becoming a stationary cluster member. DLA is a serial model: Only one gas particle diffuses at a time. A few parallel versions of DLA, with multiple particles diffusing at once, have been considered [3–6]. The parallel model we will extend begins with a uniform dilute configuration of gas particles. They aggregate, but no more are ever added to the system. Before a substantial fraction of the particles aggregate, this parallel version of DLA is equivalent to the serial version [3,4]. When a large cluster has formed, the simulated structure can be compared to structures found in nature, typically by comparing the fractal dimensions. DLA is not a thermodynamic model: Particles stick irreversibly, so there is no notion of detailed or semi-detailed balance.

The RA model extends the DLA model by placing a parallel DLA model in contact with a heat bath, implemented as a field of diffusing heat particles, or "tokens", on a superimposed lattice. Both gas and heat particles diffuse throughout the system independently, each on their own lattice, under pseudorandom dynamics. They diffuse freely over the cluster and through empty space. The gas and heat particles interact through aggregation and evaporation events along the boundary of the connected cluster.

A potential aggregation site for a gas particle is a site unoccupied by cluster, with exactly one nearest neighbor occupied by cluster. Upon reaching such a site a gas particle will aggregate, becoming a cluster particle and releasing a heat particle, which represents the latent heat of crystallization. This interaction is contingent on there being room locally in the heat bath to accept the heat particle. Explicitly modeling the latent heat released upon aggregation provides a mechanism for modeling the inverse process of evaporation. Similarly, a potential evaporation site is a site occupied only by a singly connected cluster particle. A heat particle arriving at such a site is absorbed by the cluster particle which evaporates, becoming a gas particle. This is contingent on there being no gas particle already at that site.

The model is implemented with a two phase rule. Diffusion steps, in which the particles move, alternate with interaction steps in which the gas, heat, and cluster interact, allowing aggregation and evaporation. The interaction rule is as given above. The details of the diffusion implementation are given in Sec. IV A. In order to facilitate a parallel updating of the space, we divide the system into checkerboard sublattices. Since our interaction range is nearest neighbor, we can update all the even parity sites while holding the odd parity sites fixed and vice-versa.

In our cluster growth simulations, we begin with an empty heat bath, thus only aggregation can occur. We observe rapid, nonequilibrium growth of the cluster and concomitant increase in the population of the heat bath. The occupancy of the heat bath (hence also the mass of the cluster) quickly reaches steady state, meaning evaporation and aggregation events are equally likely. At this point the system has attained a single well defined temperature, despite the fact it has not yet reached thermodynamic equilibrium. During the subsequent slow approach to thermodynamic equilibrium we observe a quasistatic annealing of the cluster. The cluster morphology, as quantified by the fractal dimension, undergoes a transition from the typical ramified pattern observed for irreversible models of diffusive cluster growth (resembling frost on a window pane), to the highest entropy macrostate allowed for a connected cluster in a finite volume: a branched polymer. Figure 1 shows the typical cluster morphology in each of the two regimes. The small grey dots also shown are the gas particles.

The dynamics of the RA model captures a number of properties of realistic microscopic physical dynamics such as locality, conservation of energy, determinism, and microscopic reversibility. Since aggregation and evaporation are both allowed and heat is explicitly modeled, any transition between two states may occur in either direction. This gives us a realistic thermodynamics: When started from a low entropy state (*e.g.,* with an empty heat bath), entropy increases and the system approaches a state of detailed balance, or thermodynamic equilibrium. Since we realistically model thermodynamic variables—local heat flow and the creation of entropy—we can do more than study

simulated structures: We have a laboratory for studying nonequilibrium thermodynamic behavior of growing clusters of particles. For a detailed discussion of the thermodynamics, the temperature, and the evolution of the growth morphology see Ref. [1].

Until now we have been discussing a closed two dimensional system with periodic boundary conditions. If we modify the boundary conditions of the heat bath lattice from being periodic to open, we essentially place the heat bath in contact with a reservoir at zero temperature and release the heat particles into this reservoir once they reach the edges of the heat bath. We can control the relative speed with which this happens through independent control of the heat and gas diffusion lengths and thus control the effective temperature in which the aggregate grows. The relative tunability of the two diffusion fields allows us to observe a rich variety of growth structures with a continuous transition from structures resembling uniform growth, through invasion percolation, to classical DLA growth.

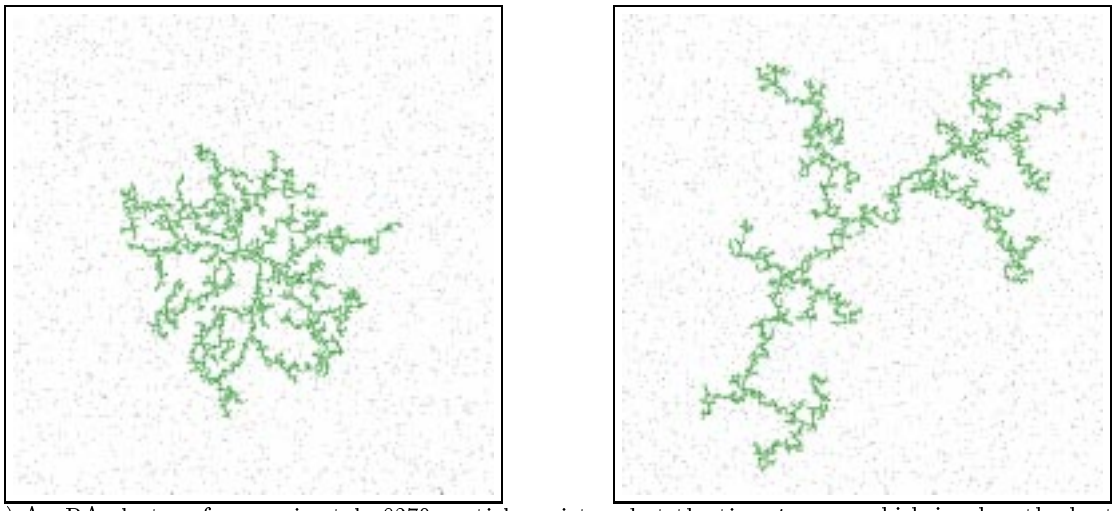

FIG. 1. (a) An RA cluster of approximately 8270 particles, pictured at the time $t = \tau_{\mathrm{T}}$, which is when the heat bath and the gas-aggregate system first reach the same temperature. Note the grey dots represent the diffusing gas particles. (b) An RA cluster with the same number of particles pictured at time $t = 80\tau_{\mathrm{T}}$. The fractal dimension for this cluster ($d_f = 1.63 \pm 0.02$) has apparently reached the asymptotic value, and is equivalent to the fractal dimension for a quenched branched polymer.

III. COMPUTATION IN REVERSIBLE SYSTEMS

Computation was long considered to be an inherently dissipative process, requiring the "decision of a two-way alternative and elementary transmittal of one unit of information" (von Neumann as quoted in Ref. [7]). A quantitative understanding of the mechanism of dissipation came with Landauer's work on erasing a bit of information [8]. Erasure requires the transfer of information from computational to other degrees of freedom and normally ultimately to thermal degrees of freedom. The lower bound on the heat produced by erasing one bit is $kT\ln 2$. More than a decade later Bennett showed that erasure is not necessary: Computation can in principle be performed with no dissipation (*i.e.*, no loss of information) [7]. Bennett's proof was based on an abstract Turing machine. He suggested RNA transcription (in the limit where the rate of transcription approaches zero), as a possible example of a dissipationless digital process. Interest in physical models of computation began around this same time, focusing on prototype Brownian motion computers such as Bennett's RNA model [9,7].

After yet another decade, Fredkin introduced the concept of conservative logic and introduced a universal conservative logic gate [10]. His goal was to formulate laws of computation more like the laws of microscopic physics, with particular emphasis on microscopic reversibility and exact conservation laws. Conservative logic gates are reversible, and the total number of one's on the wires of a closed system is conserved. The output of each gate is a permutation of its inputs. The Fredkin gate is a three input, three output conservative logic gate which implements a conditional swap. A schematic is shown in Fig. 2. The standard logical primitives ("and","not", and "fanout") can easily be built out of a Fredkin gate by supplying some constant inputs. The existence of a universal, reversible logic gate means it is possible to implement any digital computation out of these gates without ever erasing information. Moreover it makes reversible computation and circuit design look very similar to conventional computation and circuit design (but using different primitive logic elements).

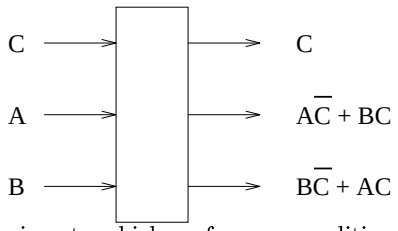

FIG. 2. The Fredkin gate: a conservative logic gate which performs a conditional swap. If the value of the signal C is true, signals A and B are interchanged; otherwise A and B go straight through. This is a reversible operation which conserves ones: the number of ones entering and leaving the gate is the same.

Fredkin was also interested in embedding his conservative logic computations into realistic physical systems. In Fredkin's Billiard Ball Model (BBM) of computation [10], finite diameter moving balls are the signals. Collisions between balls implement logic gates. The trajectories of the balls are the wires. Strategically placed mirrors reflect the balls, implementing bends in the wires. The initial positions of the mirrors and the balls must be carefully chosen to allow for precise control and synchronization of signals. The BBM Cellular Automaton (BBMCA) is a universal, reversible CA modeled after the BBM [11]. It uses pairs of particles as the "balls", with the spacing between particles corresponding to the finite diameter of the balls.

IV. COMPUTATION IN THE RA MODEL

The scheme proposed for computation in the RA model is similar to that for the Billiard Ball Model. They both use conserved particles as signals and timing and synchronization are crucial. The conditional aggregation events in the RA model are sufficiently complicated that we expect *a priori* that the model is capable of computation. We can show this capability if we allow ourselves to explicitly specify the initial state of all the microscopic degrees of freedom of the system. We must specify the initial positions of all of the gas, cluster, and heat particles, the positions of the "mirrors" which control the diffusion (as discussed below), and the parameters controlling the motion of these mirrors (as discussed in Sec. IV C 6).

We will first show how to implement wires and delays, and how to route signals. Using these elements and the RA model interaction we show the structure of a simple one-time-use logic gate, and thus prove the system is capable of computing combinatorial logic functions. However, we are interested in more general circuits: those with reusable gates, feedback, and memory elements. To this end we introduce a more complicated dyadic signaling implementation and describe a reusable universal logic gate. We then discuss issues of interconnection and signal crossing. With these in hand, we can build a simulator for any digital logic computer. We demonstrate the construction technique for a simple example circuit.

A. Signal routing and delay

In the RA model the gas and heat particles undergo pseudorandom walks, implemented using a lattice gas transport algorithm. For each particle species (gas and heat), there are two transport channels moving in opposite directions along one of the principle lattice directions. At consecutive updates of the system, we alternate between lattice directions (i.e., transport is along the $+\hat{x}$ and $-\hat{x}$ directions on odd time steps and along the $+\hat{y}$ and $-\hat{y}$ directions on even time steps). This scheme can be extended to arbitrary dimensions by introducing additional substeps along each additional lattice direction. A particle remaining in one channel exclusively will follow a diagonal path through the lattice as shown in Fig. 3(a).

To simulate diffusion we cause the particle to switch between the two channels at random. We include a pseudo-random number field: a "random" binary variable at each site at each time $\eta(\vec{x}, t)$. If $\eta(\vec{x}, t) = 1$ the particle switches channels. At the end of each update we spatially permute the η values in a deterministic and invertible manner so as to have fresh random bits at each site, while maintaining constant the probability that $\eta = 1$. The permutation must be deterministic, so we can invert the dynamics when running the model in reverse. This allows us to recover the data used to make switching decisions so we can unswitch the particles and invert the "random" walks. If we use the identity permutation, the η values remain fixed and so does the particle motion (which depends on these values).

If the pseudorandom bit fields are initially filled randomly or pseudorandomly with ones and zeros, the gas and heat particles switch between channels in an unbiased manner and simulate large scale diffusion (c.f. Ref. [1] for a quantitative discussion of diffusion coefficients and a comparison of theoretical and empirical results). If instead, the $\eta(\vec{x}, t)$ bits and their motion are precisely controlled, the one bits act as deliberately placed mirrors, switching the

gas and heat particles between transport channels at determined locations. An example with fixed mirrors is shown in Fig. 3(b). Between encounters with mirrors, the particles "stream" along a given channel uninterrupted. The gas and heat particles are controlled by separate η bits, so the gas and heat particles are reflected by separate mirrors.

Timing and synchronization are crucial to our logic scheme. To adjust timing, we can use delay loops. A delay loop can be constructed from a collection of mirrors, placed to implement a sequence of reflections. Using the transport algorithm described above, each particle takes a step in the horizontal and then the vertical directions. Since it takes at least four steps for a particle to return to its original location, delays must be a multiple of four. Figure 3(c) shows a delay loop of eight time steps.

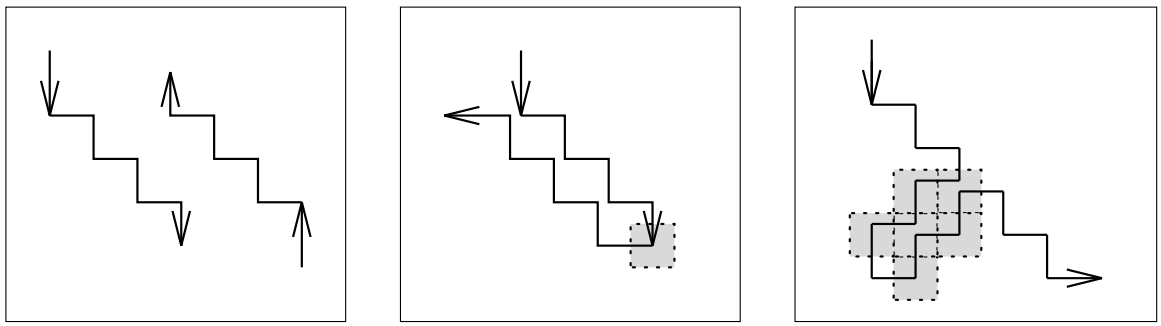

FIG. 3. (a) Streaming in channel 1 and channel 2. (b) A mirror. (c) A delay loop of eight time steps.

B. Universality: a simple gate

Consider a simple gate, as shown in Fig. 4. If we count only gas and heat particles entering and exiting at A, B, C, and D (*i.e.* not counting cluster particles), this gate conserves particle number. The shaded squares represent cluster, and positions of the mirrors are implied by the particle paths. Signal A must precede B in time. This gate has the following truth table:

A	B	C	D
gas	gas	heat	heat
gas	heat	heat	gas
heat	gas	gas	heat
heat	heat	gas	heat

If we take the signal convention that a gas particle represents one logical truth value, and a heat particle the other, then clearly this gate is universal. Say, for instance, gas represents true and heat represents false. Then $C = \bar{A}$, and $D = A\bar{B}$. So, if we can build arbitrary networks of gates such as this, then we can build arbitrary logic circuits.

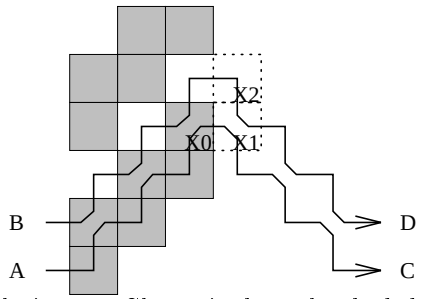

FIG. 4. A simple, non-reusable, universal logic gate. Cluster is shown by shaded squares. Signal paths are shown as wiggly arrows. Mirror locations are implied by the paths. A and B enter as shown, with A preceding B in time. If A is gas it aggregates at X1, releasing heat. If A is heat it evaporates X0, becoming gas. If A has aggregated at X1, then if B is gas it aggregates at X2, yielding heat, and if B is heat it re-evaporates X1, yielding gas. If A has evaporated X0 and B is gas, B re-aggregates X0 yielding heat, and if B is heat, it does not interact and remains heat. Recall both gas and heat particles diffuse freely over the cluster.

There is a subtlety here, however. This gate is universal, yet it is not reusable. So in fact we may only build combinatorial logic circuits, and not those with feedback. If we wish to simulate the operation of a universal Turing machine, we must "unroll" the operation of the TM. That is, we simulate the time evolution of the machine's state by computing each state combinatorially from the previous state. This spreads the time progress of the computation out spatially, requiring more logic levels for each step we wish the machine to execute. So with a polynomial amount of space, we may simulate the machine's action for a polynomial number of steps.

We can go further, though. We wish to simulate normal reusable digital logic, so that we can build arbitrary logic circuits, with feedback, memory, etc. To this end, we propose a scheme for reusing gates which utilizes matched pairs of aggregation and evaporation events. It is a dyadic signaling scheme in which pairs of particles, appropriately delayed, are routed through the same gate so as to clean up after the computation (i.e., remove the state from the gate) leaving the gate ready for reuse.

C. A reusable gate, gate interconnection, and circuits

The gate we choose to implement is the "switch gate": a two input, three output conservative logic gate which is universal [10]. It has a control input, which we will call B, and a switched input, which we will call A. The B input passes through the gate unchanged. The A input is routed to one of two outputs, conditional on the state of B. A schematic is shown in Fig. 5.

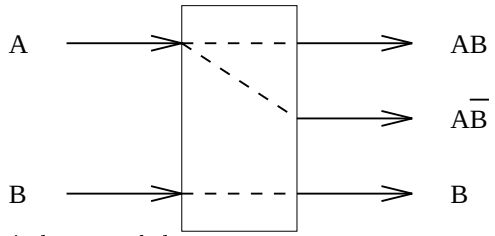

FIG. 5. The schematic diagram of a switch gate. If the input B is true, the signal A exits the top output. If B is false, A exits the middle output.

1. Summary of the gate implementation

Our first consideration in implementing such a gate is that the gate must be reusable, or stateless, as mentioned above. To realize this, we adopt a dyadic signaling convention, in which the presence of a "one" is represented not by arrival of a single particle, but by a gas particle followed four time steps later by a heat particle. This allows us to use aggregation events to implement interactions without leaving permanent changes in the structure of the gate: If a gas particle enters an input and leaves a cluster bit behind, then the corresponding heat particle follows, cleaning up the aggregation event, and leaving the gate in its original state.

We implement a switch gate in the RA model as follows. We first set up an initial condition with a few cluster bits forming a simple aggregate, with only one potential aggregation site, X1. A second site, X2, will become a potential aggregation site if and only if X1 becomes occupied by cluster. There are input paths for signals A and B. The path for signal B is routed through X1, and that for A through X2.

The action of the gate is as follows: We route the gas particle of B through X1, and delay the heat particle of B for now. If B is true (that is, the gas and heat portions of B are indeed present), the gas aggregate at X1, and X2 now becomes a potential aggregation site. If B is false, nothing happens and X2 is not a potential aggregation site. Now we arrange, by appropriate choice of delays, for both particles of A to pass through X2 while X1 is (in the $B = 1$ case) occupied by cluster. If B is false, A passes through X2 unchanged (*i.e.* with its heat portion following its gas portion). If B is true, the gas portion of A aggregates at X2 yielding heat, and the heat portion of A evaporates the cluster particle at X2, yielding gas again. Thus, B being true essentially reverses the order of gas and heat particles exiting X2. When both the gas and heat portions of A have had a chance to interact at X2, we send the heat portion of the B signal through X1. If B is a one, the heat signal encounters the cluster particle at X1 and evaporates it, thus restoring the connected cluster to its original state and leaving the gate ready for reuse. Note if B is false nothing happens, which also leaves the gate ready for reuse.

The particles for A exit X2 in the same direction regardless of the state of B, but in different temporal orders. To obtain the two outputs $A\bar{B}$ and AB on different spatial paths, we use η fields which change with time to switch the gas and heat particles onto two different paths according to their timing. We then delay the heat on the AB

output path, to conform with our dyadic signaling convention that heat follows gas. A schematic diagram of this interdependence of events is shown in Fig. 6.

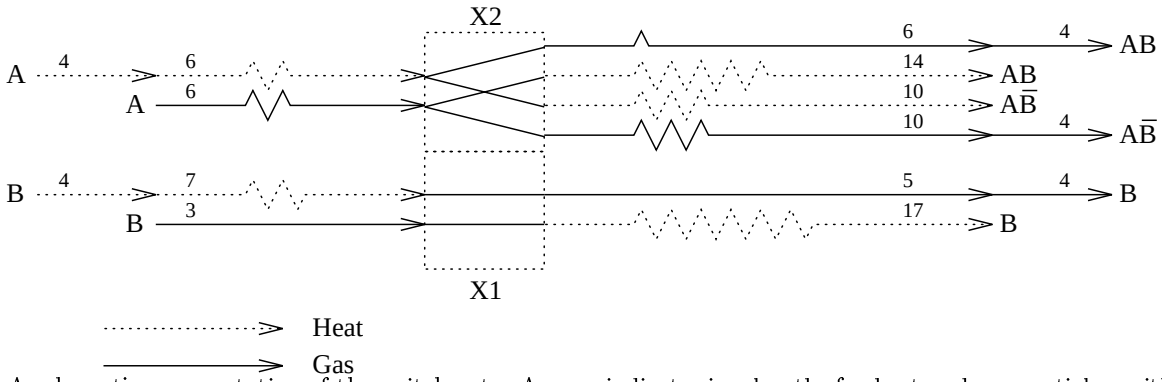

FIG. 6. A schematic representation of the switch gate. Arrows indicate signal paths for heat and gas particles, with delays annotated. Signals A and B enter simultaneously from the left at time zero. The heat particle for each signal trails the gas particle by four time steps. If B is true, X1 is occupied from time three through time 11. In this case, A aggregates and evaporates from site X2 at times six and ten respectively. The signal paths are indicated by the top two lines in X2. If B is false, X1 is never occupied. In this case, both A and its cleanup signal pass through without interacting, and exit at the appropriate times on a different path. The signal paths in this case are indicated by the bottom two lines in X2. Output delays are chosen to place the signals at the gate outputs at a time independent of input values and signal path. Note that all paths have an identical length of 20.

2. Interconnect and parity

There are some subtleties involved in being able to route any output of any gate to any input of any other gate. In particular, there is a parity defined on particles: Since they are constrained to move alternately vertically and horizontally, we may draw a checkerboard on our lattice and separate the particles into those starting at time zero on a red square ("red" particles), and those starting on a black square ("black" particles)[1]. Since the first move is horizontal for all particles, any red particle will move horizontally off its red square, and will always move horizontally off any red square. Similarly any black particle will always move horizontally off a black square. These types of particles cannot be interconverted, and a path followed by one type cannot be followed by the other. Each gate input follows a specific path, thus requiring a particular color particle. Likewise, each gate output produces a particular color particle. So if we had a gate output producing a black particle and another gate's input expecting a red particle, there would be no way of directly connecting that input to that output. To solve this, we constrain all our inputs and outputs to be on black squares of the checkerboard.

Even with all inputs and outputs on black squares, we have a synchronization problem: Our gates require synchronous signal arrivals, so sometimes we need to delay individual signals to satisfy this requirement. Note delay loops are only available in multiples of four time steps (see Fig. 3). Consider any two input signals to our circuit. If they are ever to interact in any gate, they must not only start on the same color square, but they must also be routable to each other in a multiple of four time steps; otherwise they will never be able reach the inputs of the gate simultaneously. So we introduce an additional constraint that all inputs to our circuit be routable to each other in multiples of four time steps. This constraint can be met simply by placing all inputs at time zero on a sublattice of a checkerboard: a square sublattice with separation distance two (see Fig. 7). All inputs are now separated by paths whose lengths are multiples of four, and in fact particles can travel from one site of the sublattice to an adjacent one in exactly four time steps. This also guarantees that all outputs of our circuit will be routable to each other in multiples of four time steps (since gate departure times as well as arrival times are synchronous). Note that by restricting our inputs to be on this sublattice, we have somewhat reduced the usable volume of space. This simplifies our discussion, however, and introduces only a constant factor slowdown and size increase.

---

[1]Note, this discussion is independent of the checkerboard updating scheme discussed in Sec. II.

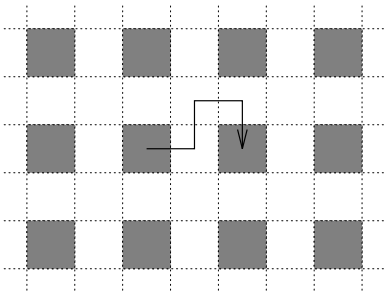

FIG. 7. The RA lattice, with the circuit input/output sublattice shown in shaded squares. A particle is shown traveling from one sublattice site to an adjacent one in four time steps.

3. Details of the switch gate implementation

The details of the RA model switch gate are shown in Fig. 8, which depicts the initial state of the lattice and the paths taken by the signals. As mentioned above we set up an initial condition with a few cluster bits forming a simple aggregate. X1 and X2 are indicated by heavy dashed lines. One can see that X1 is initially a potential aggregation site, whereas X2 is a potential aggregation site if and only if X1 is occupied by cluster. The "extra" cluster bits prevent spurious aggregation at undesired locations.

The paths taken by the particles are shown as the wavy lines, and are determined by the placement of mirrors (indicated by the various shades of gray as shown in the legend). At the left are the two input signal paths; at the right are the three outputs. Note that all inputs and outputs lie on the sublattice described in Sec. IV C 2, thus guaranteeing routability.

One can verify by "walking through" the paths by hand, that the actual delay times are as specified in Fig. 6. If B is true, X1 is occupied between times three and 11. In this case, if A is also true, a heat particle exits X2 at time six, and a gas at time ten. If B is false and A is true, a *gas* particle exits X2 at time six and a *heat* at time ten. The gas and the heat so produced travel to a toggling mirror, arriving at times ten and 14 respectively. The toggling mirror begins by deflecting gas in the down direction and heat in the up direction at time zero, and toggling gas and heat directions every four steps thereafter. Hence, if B is true and a heat arrives at time ten and a gas at time 14, they are both deflected in the up direction, since the mirror toggles in between them at time 12. Conversely, if B is false, the gas arriving at time ten and the heat arriving at time 14 are both deflected *downward*, again since the mirror toggles at time 12. The extra eight steps of heat delay in the upper exit path are to delay the heat so that it follows the gas, in accordance with our signaling convention.

The longest path through the gate requires sixteen time steps, so we have imposed extra delay on some signal paths so that the propagation delay through the gate is exactly sixteen on all paths. This will be convenient when building more complex circuits, since we will not have to worry about delaying signals to compensate for differences in gate delay. The paths in the figure are longer than the gate propagation time so that both the gas and heat particles can be shown entering and exiting the gate.

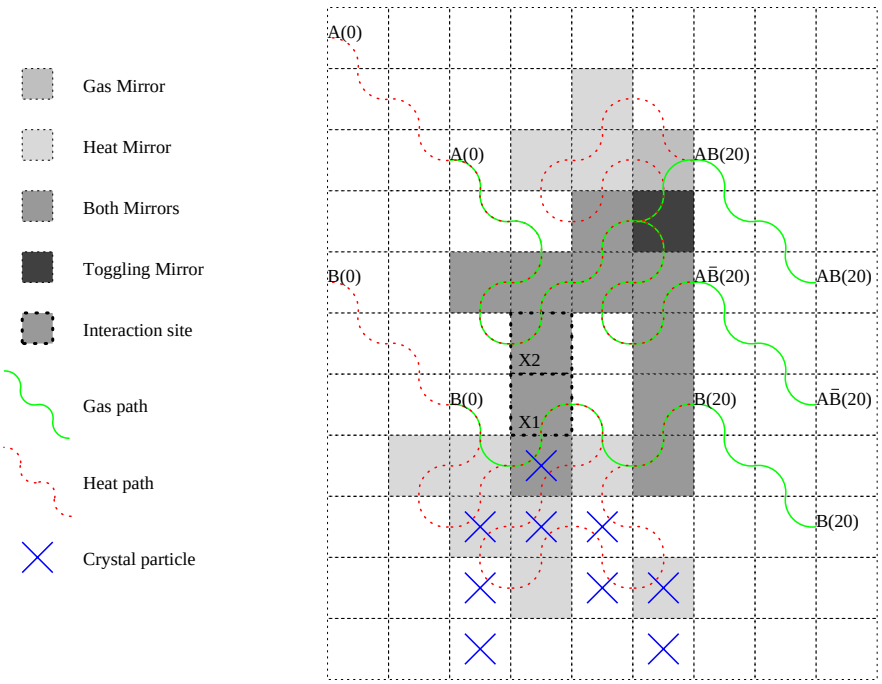

FIG. 8. A detailed picture of an RA switch gate. Signal entry and exit times are given in parentheses after the signal name.

4. The reverse switch gate

In order to build general logic circuits we will need to use switch gates in both the forward and reverse directions [10]. The switch gate used in reverse is a three input two output gate which performs the inverse logic operation. The three inputs must of course be appropriately correlated for this to be possible. Since the dynamics of the RA model is reversible, we can build the reverse gate by sending the signals through the outputs of the original gate, but with the opposite signaling convention: heat preceding gas. Converting between forward and reverse signaling conventions is simple: We need only impose an extra delay of eight steps on the gas so that it follows the heat by four time steps. We may convert back to the forward signaling convention by similarly delaying the heat.

5. Crossing wires

We showed above that, observing parity constraints in gate placement, any output can be connected to any input. However we have yet to show that signals can cross—a necessary detail for connecting any output to any input. Consider two signals, one traveling in a diagonal downward and to the right, the other traveling in a diagonal upward and to the right. The first signal will travel purely in one channel (channel 1 in Fig. 3(a)), encountering no mirrors. The second signal must flip channels at each time step and thus encounter a mirror at every site. Thus for these signals to cross there must both be mirrors and be no mirrors at the two lattice sites they will both encounter. A way to implement this is to delay one signal by four time steps, allowing the first signal to propagate past the relevant two sites, then toggle the mirrors. The second signal now encounters the correct mirror configuration when it occupies the relevant two sites. A "cross gate" so constructed is shown in Fig. 9. The shaded sites are the ones where the mirrors are toggled. Note that we have delayed the first signal by four time steps after its encounter with the toggling sites, so that both signals leave the cross gate synchronously.

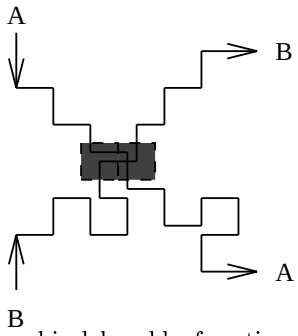

FIG. 9. A "cross gate". The signal traveling upward is delayed by four time steps while the signal traveling downward passes through the two shaded lattice sites. The mirrors at the shaded sites are then toggled. The upward signal now encounters the correct mirror configuration to pass through as indicated. The first signal is delayed by four steps after passing through the shaded sites so that both signals leave this gate synchronously.

6. A sample circuit

The RA model has been implemented on a special purpose cellular automata machine, the CAM8 [12]. A detailed description of this implementation can be found in Ref. [1]. For constructing the logic gates we have discussed in this paper, we use the CAM8 implementation of the RA model, but with the modification that the initial state and the dynamics of the pseudorandom bits are precisely specified. In the original model the permutation of the η bit fields was simply a displacement (shift) in $\hat{x}$ and $\hat{y}$ by a prespecified amount at each update step. Here, where we wish to exercise detailed control over the microscopic dynamics, we choose the displacements more restrictively. We displace the η bit fields for both gas and heat by half the size of the space in the horizontal direction only. Further, this displacement occurs only on every fourth time step. This allows us to implement the toggling mirrors easily. We place all the circuitry and one set of mirrors in the left half of the space and place a second set of mirrors, with the toggling mirrors complemented, in the right half. Note that since the shifts of the pseudorandom bit planes are specified as part of the initial condition, we are still just manipulating our initial condition in order to effect computation in the RA model.

To see the detailed action in the CAM8 simulation, consider a single switch gate. The operation of this gate in three of the four input cases is shown in Fig. 10. Note that A takes the topmost output path if and only if $B = 1$.

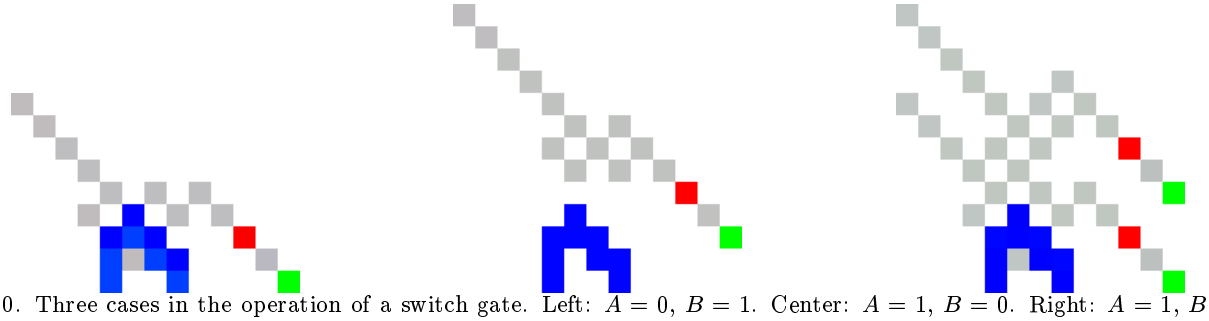

FIG. 10. Three cases in the operation of a switch gate. Left: $A = 0$, $B = 1$. Center: $A = 1$, $B = 0$. Right: $A = 1$, $B = 1$. Particles are heavily shaded, cluster is black, and paths are lightly shaded. The path shading is only for every alternate time step, to make it simpler to resolve distinct paths by eye. The null case, $A = 0$, $B = 0$ is omitted.

We have shown how to implement a reusable, universal logic gate, how to route and delay signals, and how to let signals cross so that output from any can serve as input to any other gate. Thus we can build any Boolean logic circuit we wish. As a simple example of connecting gates together we construct an identity circuit from two back to back switch gates, the second running in reverse, followed by a cross gate. The schematic of the circuit is shown in Fig. 11. The CAM8 implementation is pictured in Fig. 12. The black squares are the stationary cluster bits. The lightly shaded squares mark the trace of where the signals have traveled (the wires). The signals (heavily shaded squares) exit the gate at the right. Note that in the space between the two switch gates, the gas particles are delayed by eight so the signals enter the reverse gate with the opposite signal convention. In order not to clutter the figure we did not reinvert the signals, so they leave with the inverse signal convention of heat first.

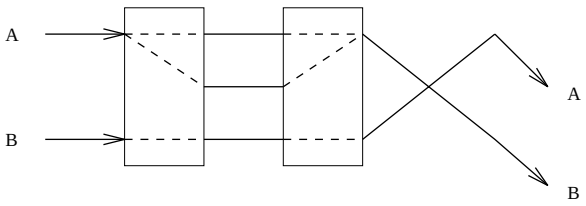

FIG. 11. A circuit composed of two switch gates—one forward, one reverse—followed by a signal crossover.

FIG. 12. A CAM8 implementation of an identity gate (composed of two switch gates back to back) and a crossover. The shaded squares are the wires, visualized every other time step. The signals have exited the circuit at the right.

V. DISCUSSION AND CONCLUSIONS

A. Summary

We have investigated the computational capabilities of a specific physical model of cluster growth, where we have control over all of the microscopic degrees of freedom. The dynamics can compute any sequence of digital logic operations if the system is initialized to a precisely specified state. Thus the RA model can simulate any other digital dynamics.

The moving particles are the logic signals, the paths they take are the wires. Synchronization of signal arrival times at specific sites requires careful layout of mirrors which route and delay signals. By routing particles through specified interaction sites, we can build logic gates from conditional aggregation and evaporation events. These constructs are relatively straightforward to implement, allowing us to build a simple logic gate as shown in Sec. IV B. This gate is changed by the interaction—it can be used only once. Showing that the RA model can support reusable gates adds complication. We have many degrees of freedom; choosing an appropriate signaling convention is crucial. We choose a dyadic signaling convention which allows us to construct reusable logic gates, as discussed in Sec. IV C 3.

B. Computation in real physical systems

We have exhibited computations in a discrete lattice system that depend on exactly synchronous updating and complete control of all microscopic degrees of freedom. Have we abstracted anything useful for computation in real physical systems?

The RA model captures some essential aspects of real cluster growth. Perhaps the most relevant to computation is the conditional nature of aggregation (*i.e.*, the presence of nucleation sites only at the perimeter of a growing cluster along with the absence of heat in the local environment). This conditional interaction allows us to build logical primitives that do not depend on exact synchronization.

Computation in the RA model can be accomplished with exact control of the microscopic degrees of freedom. However, in real physical systems, including electronic computers, we typically have control only of the macroscopic degrees of freedom. For the macroscopic dynamics of a system to be universal, the microscopic dynamics must necessarily be universal: If we cannot compute with complete control over the system, we cannot hope to compute with less control. Thus we can consider the construction given in this paper as a "warmup" for addressing the "larger" issue of macroscopic universality. The analogy to thermodynamics is straightforward. A gas can do mechanical work moving a piston despite the fact we know nothing of the individual gas particles; We know only aggregate quantities. The question then is, can we get *computational* work out of a system with control only of the macroscopic degrees of

freedom? With this type of understanding we might be able to compute with a growing bacterial colony or a growing crystal aggregate, and exert detailed control over the structures produced [13].

A first step in understanding macroscopic computation in the RA model is to test the robustness of the system's ability to compute when it is subjected to an actual stochastic dynamics. For instance we can study the situation where the particles follow truly random walks or where the interactions are probabilistic, yet we maintain control over the other degrees of freedom. If we replace each single gas and heat particle with a dilute cloud of particles we may be able to conditionally aggregate and evaporate with high probability. Directing the macroscopic motion of clouds of particles may require us to add momentum conservation to the RA model, making it even more realistic. Even so, restoring the gates and signals to their starting state would require some dissipative cleanup process. This would entail a constant throughput of "power" (*i.e.*, the addition and removal of particles).

ACKNOWLEDGMENTS

This work was supported by DARPA under contract number DABT63-95-C-0130, and by the Merck/MIT Graduate Research Fellowship Program. We wish to thank the organizers of the SFI Constructive CA workshop, which provided inspiration for this work. We also thank Jeremy Zucker for a careful reading of this manuscript.

---

[1] R. M. D'Souza and N. H. Margolus. A thermodynamically reversible generalization of Diffusion Limited Aggregation. Available as cond-mat/9810258, To appear in *Phys. Rev. E.*, 60(1), 1999.

[2] T. A. Witten and L. M. Sander. Diffusion-Limited Aggregation, a kinetic critical phenomenon. *Phys. Rev. Lett.*, 47(19):1400–1403, 1981.

[3] R. F. Voss. Multiparticle fractal aggregation. *J. Stat. Phys.*, 36(5/6):861–872, 1984.

[4] T. Nagatani. Unsteady diffusion-limited aggregation. *J. Phys. Soc. Jpn.*, 61(5):1437–1440, 1992.

[5] H. Kaufman, A. Vespignani, B. B. Mandelbrot, and L. Woog. Parallel Diffusion Limited Aggregation. *Phys. Rev. E*, 52:5602–5609, 1995.

[6] K. Moriarty, J. Machta, and R. Greenlaw. Parallel algorithm and dynamic exponent for diffusion-limited aggregation. *Phys. Rev. E*, 55(5):6211–6218, 1997.

[7] C. H. Bennett. Logical reversibility of computation. *IBM Journal of Research and Development*, 17:525–532, 1973.

[8] R. Landauer. Irreversibility and heat generation in the computing process. *IBM Jour. Res. Dev.*, 3:183–191, 1961.

[9] R. W. Keyes and R. Landauer. Minimal energy dissipation in logic. *IBM Journal of Research and Development*, 14:152–157, 1970.

[10] E. Fredkin and T. Toffoli. Conservative logic. *Int. J. of Theor. Phys.*, 21(3/4):219–253, 1982.

[11] N. H. Margolus. Physics-like models of computation. *Physica D*, 10:81–95, 1984.

[12] N. H. Margolus. CAM-8: a computer architecture based on cellular automata. In A. Lawniczak and R. Kapral, editors, *Pattern Formation and Lattice-Gas Automata*. American Mathematical Society, 1996.

[13] H. Abelson, T. F. Knight, Jr., G. J. Sussman, and friends. Amorphous computing, MIT AI Laboratory. *http://swissnet.ai.mit.edu/ switz/amorphous/white-paper/amorph-new/amorph-new.html*, 1995-present.