Phase transition in a swarm algorithm for self-organized construction

Henri Leung
Complex Adaptive Systems Laboratory, ECECS Department, University of Cincinnati, Cincinnati, Ohio 45221-0030, USA

Ravi Kothari
IBM—India Research Lab, Block I, IIT Hauz Khas, New Delhi 110016, India

Ali A. Minai
Complex Adaptive Systems Laboratory, ECECS Department, University of Cincinnati, Cincinnati, Ohio 45221-0030, USA

(Received 20 December 2002; revised manuscript received 10 June 2003; published 13 October 2003)

This paper reports on a system where very simple, noncommunicating mobile agents in a cellular (lattice) environment use purely local rules to construct connected structures from initially randomly distributed building blocks. We study the effect of block density on the final structure, demonstrating a percolationlike phase transition: Low block densities lead to the formation of small, disconnected structures but a single connected structure emerges abruptly beyond a critical density. The empirical study of the structure at the transition point shows scaling behavior, providing strong evidence for criticality. We also demonstrate that a simple change of rules can completely change the phase-transition effect. The results have implications for the self-organized construction of complex structures by swarms.

DOI: 10.1103/PhysRevE.68.046111 PACS number(s): 05.65.+b, 89.75.Da, 89.75.Fb, 45.70.Qj

I. INTRODUCTION

Recently, there has been considerable interest by physicists, biologists, computer scientists, and engineers in the study of swarm-based algorithms for a variety of applications [1], including the formation of emergent structures [2–6]. The swarm approach, inspired by insect colonies, seeks methods by which simple, locally informed behavior by a large number of simple agents can lead to the emergence of large-scale, complex organization. Such self-organization is thought to underlie the construction of structures such as termite and ant nests [2–6], the formation of near-optimal foraging patterns in ants [7,8], and the efficient and flexible task allocation seen in many insect colonies [9,10]. Swarm-based algorithms have several features that make them attractive for new technologies such as smart materials, collective and reconfigurable robots, self-assembling structures, and adaptive sensor networks. The algorithms are inherently robust, scalable, flexible, and easily extended. Also, by using very large numbers of simple (and therefore cheap) agents rather than a few complex (and expensive) ones, swarm algorithms promise economies of scale that will be necessary for many of the aforementioned technologies. Our results are especially relevant in the area of collective robotics [11–14], where large numbers of autonomous robots act cooperatively to accomplish tasks such as search, exploration, mapping, and construction.

In this paper, we report on a simple swarm technique for the construction of connected, locally linear structures by ‘‘myopic’’ agents, i.e., agents that are only aware of system state in their immediate spatial vicinity and do not consider long-term payoff in their decision-making. While of little obvious utility in their current abstract and simplistic form, such structures can form the basis of more useful ones such as might be needed by groups of robots engaged in the construction of sensing, communication, or traffic networks. Such random structures may also have potential use as scaffolding in the swarm-based construction of more complex structures. Here, we focus on the abstract problem and report on some interesting characteristics of the emergent structures produced by the algorithm. In particular, we show that the system exhibits a phase transition and the accompanying scaling behavior—a fact that may have implications for the success or failure of swarm-based construction algorithms in various settings.

II. BACKGROUND

Pattern formation is one of the classic problems studied in the swarm literature [1]. Rauch *et al.* [15] showed that antlike agents following simple rules for depositing and following an evaporating pheromone could, under some conditions, create networks of paths such as those seen in real ant colonies. They also showed that the model system underwent sharp bifurcations (phase transitions) as some of its parameters were changed. Other work on swarm-based construction has also shown that the emergent structures can depend fundamentally and qualitatively on system parameters [7,5]. In the present work, we show another example of such transition in the context of an extended spatial structure. By choosing an especially simple type of structure, we are able to use the framework of percolation theory [16] and scaling [17,18] to show quantitative evidence of structural phase transition in our system. Classical percolation theory has been concerned mainly with static random structures, whereas the structures we study are built over time by agents following rules. In this sense, the present work is much more akin to the other studies of scaling in emergent structures such as networks [19], bacterial colonies, and material aggregates [20]. It is also closely related to—and relevant for—the currently active area of amorphous computing [21], where computational structures are built via self-organization [21–24].

There is also a significant body of work on the self-organization of robot swarms into regular formations such as lines, circles, and polygons [25–27]. Our work is peripherally related to this since we consider the patterns constructed by agents using building materials rather than relationships among the agents themselves. Indeed, as described below, our agents are oblivious to other agents and all communication in the system occurs through the pattern being constructed. In the swarm and collective robotics literature, this is known as *stigmergy* [11], and is believed to underlie most construction in insect colonies [1]. Stigmergy has several desirable features from an applications viewpoint: It is inherently decentralized and, therefore, scalable; it works with simple agents that need no communication devices or infrastructure, reducing cost; and it is unobtrusive, since no communication traffic is generated. The main drawback is that, by giving up the option to communicate, agents can no longer engage in explicit coordination and planning. However, in truly large-scale swarms or truly simple agents, bandwidth and cost constraints would make stigmergy the only feasible option—perhaps in combination with diffusive signaling.

III. SYSTEM DESCRIPTION

The problem we consider is *the self-organized construction of connected, locally linear structures by locally aware, noncommunicating mobile agents following simple rules.*

The system comprises *n agents* that move on a $L \times L$ grid, picking up and depositing *blocks* of material based on the state of their local neighborhood. The total number of blocks in the system is fixed, and is denoted by *N*, giving a *block density* $D \equiv N/L^2$, which is the primary parameter of interest. The set of all blocks is denoted by $B = \{b_j\}$. Initially, the blocks and agents are distributed randomly over the grid. If an agent occupies a grid cell with a block, it can *pick* up the block. If an agent carrying a block is in a grid cell with no block, it can *deposit* the block it is carrying. Both pickup and depositing are governed by rules as described below. The objective is to arrange the blocks such that they form a connected, locally linear structure covering the environment as evenly as possible. However, because the rules are triggered purely by the agents' local view, the global characteristics of the final block distribution must be emergent rather than the result of a global optimization process.

Each cell (x,y) in the environment has a *block occupancy state* $\omega_{x,y}(t)$, which is 1 if the position has a block and 0 otherwise. Cell (x,y) also has an *agent occupancy state* $\xi_{x,y}(t)$, which is an integer indicating the number of agents in cell (x,y). The eight cells surrounding (x,y) are termed its *neighborhood*, denoted by $N(x,y)$.

A block b_j in cell (x_j, y_j) is said to be a *neighbor* of a block b_k in cell (x_k, y_k) if $(x_j, y_j) \in N(x_k, y_k)$ [which also implies that $(x_k, y_k) \in N(x_j, y_j)$]. This relationship is denoted by $b_j \leftrightarrow b_k$.

A block b_j in cell (x_j, y_j) is said to be *connected* to a block b_l in cell (x_l, y_l) if (1) $b_j \leftrightarrow b_l$ or (2) $\exists$ $b_k \in N(x_j, y_j)$ such that b_k is connected to b_l. If b_j is connected to b_l then b_l is also connected to b_j. The relationship is denoted by $b_j \Leftrightarrow b_l$. Note that connectedness is also transitive, so $b_j \Leftrightarrow b_k$ and $b_k \Leftrightarrow b_l$ implies $b_j \Leftrightarrow b_l$.

Each agent *i* has a *loading state* $s_i(t) \in \{0,1\}$ at time *t*, where $s_i(t) = 0$ indicates that the agent is not carrying a block and $s_i(t) = 1$ indicates that it is. The agent's position at step *t* is given by $l_i(t) = (x_i(t), y_i(t))$.

The emergent structure is evaluated for its size and connectivity. In particular, a *cluster* B^k within the structure is defined as a set of blocks $\{b_j^k\}$ such that (1) every block in B^k is connected to every other block in the set, and (2) no block in the set is connected to any block not in B^k. The *size* of cluster B^k equals the number of blocks in it, and is denoted $|B^k|$. Note that, at any time, the set of all clusters forms a partition of *B*: $\cup_k B^k = B$, and $B^{k_1} \cap B^{k_2} = \Phi \ \forall \ k_1 \neq k_2$. The size of the largest cluster at time *t* is denoted by $M(t)$, and the size of the largest cluster in the final structure is denoted by *M*. We study the quantity $\sigma = M/N$—the fraction of blocks that are included in the largest final cluster—and show that, under certain conditions, σ shows a phase transition at a critical block density.

A. Rules for pickup and deposit

We assume that, at any time, an agent can observe the distribution of blocks in a 5×5 square neighborhood centered on its current position. However, most rules only use a 3×3 neighborhood, and the full range of observation is used only in a few special cases. Rules for pickup and deposit are defined using local neighborhood maps of block distributions that would trigger the corresponding action. The *three-view* $v_i^3(t) \in \{0,1\}^9$ of agent *i* at time *t* is the occupancy state of its 3×3 neighborhood. Each bit of $v_i^3(t)$ corresponds to a cell in the neighborhood, with a 1 indicating the presence of a block. The first bit $v_{i_0}^3 \equiv V_i(t)$ corresponds to the agent's current location, and the remaining bits are indexed clockwise starting from the north bit. The *five-view* $v_i^5(t)$ is defined similarly for the agent's 5×5 neighborhood, with the inner ring indexed before the outer ring. Note that $v_{i_0}^3(t) = v_{i_0}^5(t) \equiv V_i(t) = \omega_{l_i(t)}(t)$; this is termed the occupancy state of the agent's current position.

1. Agent movements and decisions

Agents move asynchronously and independently, with two agents allowed to occupy the same location simultaneously. To ensure that all agents are updated regularly while maintaining asynchronicity, the following update procedure is used.

An *update cycle*, indexed by *T*, is defined as a pass through the entire agent population, updating the state of each agent based on the *current* state of the environment and other agents, i.e., including previous updates within the current update cycle. Thus, an update cycle consists of *N* time steps, indexed by $\tau = 1, 2, \ldots, N$. Agents are selected for update *in random order* during each update cycle. This ensures that every agent is updated at least once in any window of $2N - 1$ update steps without creating fixed order effects.

Overall, time in the system is indexed by t, with $t=(T-1)N+\tau$. An agent i selected for update at step t is termed a *live agent*, and goes through the following steps in sequence.

(1) Pickup decision: If the agent is not carrying a block, i.e., $s_i(t-1)=0$, and its current position $l_i(t)$ is occupied by an unpicked block, i.e., $\omega_{l_i(t)}(t-1)=1$, it makes a decision on whether to pick up the block. If the decision is to pick up,

$$s_i(t)=1,\quad \omega_{l_i(t)}(t)=0$$

else

$$s_i(t)=s_i(t-1),\quad \omega_{l_i(t)}(t)=\omega_{l_i(t)}(t-1).$$

If the agent is already carrying a block, i.e., $s_i(t-1)=1$, there is no possibility of pickup. In that case,

$$s_i(t)=s_i(t-1),\quad \omega_{l_i(t)}(t)=\omega_{l_i(t)}(t-1).$$

(2) Deposit decision: If the agent is carrying a block, i.e., $s_i(t-1)=1$, and its current position $l_i(t)$ is empty, i.e., $\omega_{l_i(t)}(t-1)=0$, it makes a decision on whether to deposit its block at the new position. If the decision is to deposit,

$$s_i(t)=0,\quad \omega_{l_i(t)}(t)=1$$

else

$$s_i(t)=s_i(t-1),\quad \omega_{l_i(t)}(t)=\omega_{l_i(t)}(t-1).$$

If the agent is not carrying a block, i.e., $s_i(t-1)=0$, there is no possibility of deposit. In that case,

$$s_i(t)=s_i(t-1),\quad \omega_{l_i(t)}(t)=\omega_{l_i(t)}(t-1).$$

(3) Move decision and action: The possible moves for agent i are to the eight positions adjacent to $l_i(t)$. The possible directions of movement comprise the set $m=\{m_k\}=\{1,\ldots,8\}$, indicating directions clockwise with $m_k=1$ for the cell to the north. The agent chooses direction $d_i(t)\in m$ and moves to the corresponding position, which is denoted by $l_i(t+1)$. The variable $d_i(t)$ is termed the *move direction* for the agent at time t.

The choice of $d_i(t)$ depends on the agent's state, as described below.

For agents j not selected for update at time t, $s_j(t)=s_j(t-1)$ and $d_j(t)=d_j(t-1)$.

Each agent has two types of memory: a *direction memory* and a *counter memory*. At the beginning of the simulation, both memories are empty. The direction memory is $d_i(t)$, the direction of the agent's last move, and the counter memory $c_i(t)$ keeps track of the number of steps since the agent last encountered an empty cell.

2. Summary of the algorithm

Before giving a formal description of the algorithm, we give a brief qualitative description.

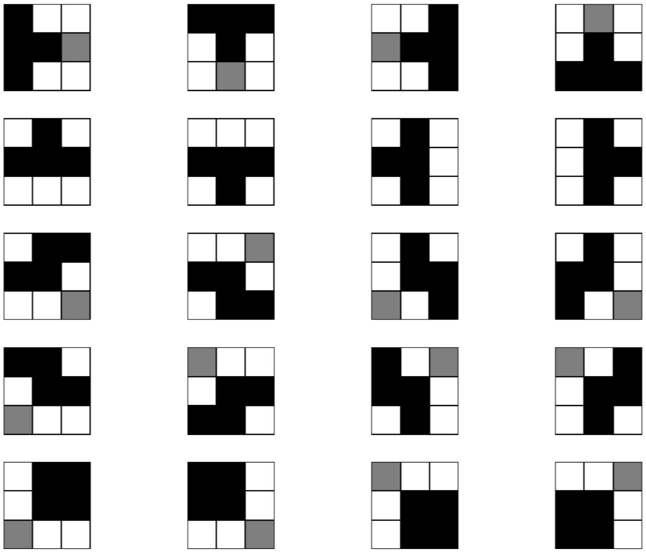

FIG. 1. Maps triggering pickup to inhibit blob formation.

The overall goal—not explicitly known to agents—is to arrange the blocks in a connected meshlike structure. The agents focus on placing blocks so as to form locally linear structures. The linear structures should not have any sharp corners except at segment junctions. To this end, they (1) pick up blocks that are isolated or are redundant for local connectivity, and (2) deposit blocks to extend existing locally linear structures they come across. An agent coming upon a block picks it up if it is part of one of the disallowed local arrangements of blocks. It then carries the block until it comes upon a connected linear filament of blocks. It follows this filament until the end and then extends it by depositing its own block. It then moves off to look for another block. These rules are described formally in the following three sections.

3. Rules for pickup

Pickup is considered only when agent i is not carrying a block and encounters a block at its current site.

The pickup decision can be expressed in terms of six stimulus-response rules. The triggering conditions for the rules are checked in the order given and the first condition to be satisfied produces the agent response. Conditions following this do not then need to be checked. Figures 1–4 show the triggering conditions for block pickup as neighborhood maps. In the maps, the grid position being considered for pickup is at the center, which is also where the agent is located. The conventions used for triggering conditions are as follows.

(1) Figure 1: To satisfy a triggering condition, black squares must be 1 (occupied), gray squares must be 0 (unoccupied), and white squares may be in any occupancy state ("don't care").

(2) Figure 2: To satisfy a triggering condition, black squares must be 1 (occupied), at least one of each triplet of gray squares must be 1 (occupied), and white squares may be in any occupancy state ("don't care").

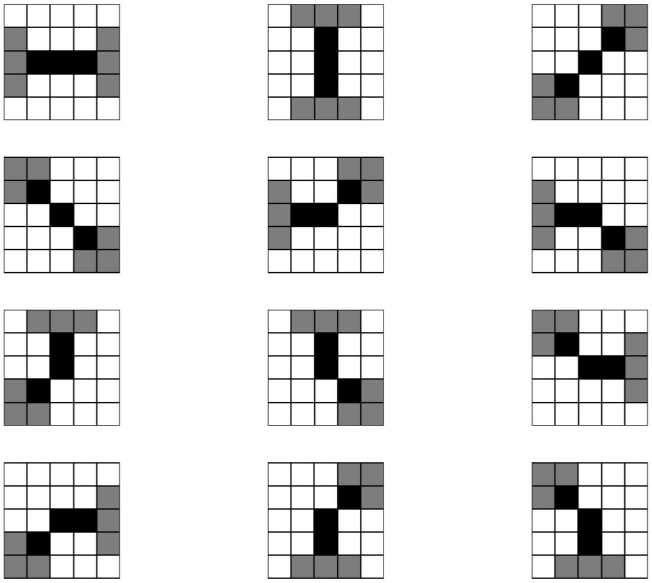

FIG. 2. Maps suppressing pickup to promote formation of long block segments.

(3) Figure 3: To satisfy a triggering condition, black squares must be 1 (occupied) and white squares may be in any occupancy state (“don’t care”).

(4) Figure 4: To satisfy a triggering condition, black squares must be 1 (occupied), at least one of each triplet of gray squares must be 1 (occupied), and white squares may be in any occupancy state (“don’t care”).

It should be noted that all the triggering conditions are based on *local* views, which is an important aspect of self-organization. The following rules are applied in sequence, and the first one to be triggered is accepted. The rules following the triggered rule are then not applied. The rules are as follows.

(1) If the block at $l_i(t)$ is isolated [i.e., $v_{i_k}^3(t)=0$ for $k=1,\ldots,8$], the agent picks up the center block.

(2) If $v_i^3(t)$ corresponds to one of the maps in Fig. 1, the agent picks up the center block. The purpose here is to discourage the formation of blobs—groups of blocks with redundant connectivity.

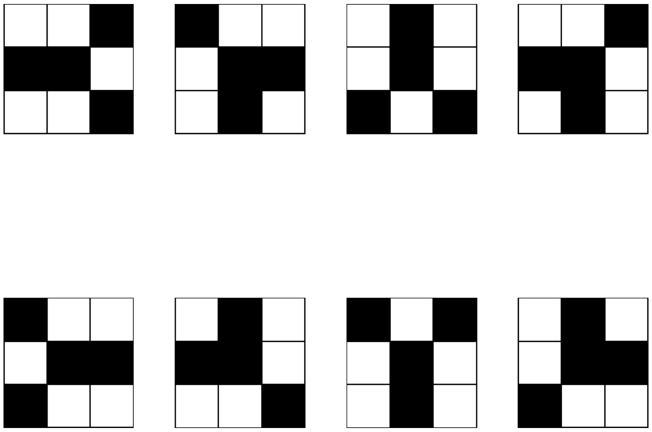

FIG. 3. Maps suppressing pickup to promote formation of block segment junctions.

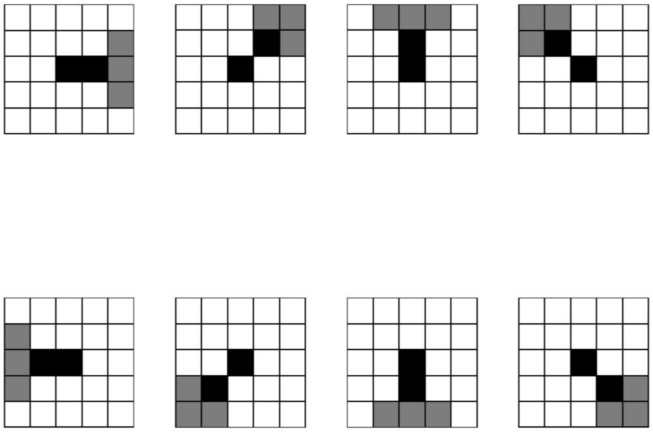

FIG. 4. Map configurations indicating segment terminals. The terminal block is picked up with probability α.

(3) If $v_i^5(t)$ corresponds to one of the maps in Fig. 2, the agent does not pick up the center block. Note that, according to the convention mentioned above, a triggering configuration in Fig. 2 is satisfied only when at least one block is present in each of its two gray cell triplets. The purpose here is to encourage the formation of long block segments that do not have any sharp corners. This will reduce the chance of loops being formed and therefore encourage the block segments to span the entire map even at low densities.

(4) If $v_i^3(t)$ corresponds to one of the maps in Fig. 3, the agent does not pick up the new block. The purpose of this rule is to encourage the formation of appropriately efficient (i.e., nonredundant) block segment junctions.

(5) If $v_i^5(t)$ corresponds to one of the maps in Fig. 4, the agent does not pick up the center block with probability $1-\alpha$ and picks it up with probability α, where α is a fixed parameter. After the exclusions specified by Figs. 1–3, the configurations in Fig. 4 are triggered only when the central block is the last or the last-but-one block on a segment. Thus, when $\alpha>0$, terminal blocks tend to get picked up. Terminal blocks of all segments can be picked up, but such blocks represent a larger fraction of very short segments, and these are disrupted preferentially. Thus, $\alpha>0$ suppresses the proliferation of short segments. The α parameter is of primary interest in the work reported here.

(6) Agent picks up the block in all the other situations.

The intuition behind the rules triggering and suppressing pickup is the following. Fig. 2 rules suppress pickup when the current block could form part of a viable line. However, the rules are sufficiently redundant that they also suppress pickup in some situations that produce bloblike structures. The pickup triggers in Fig. 1 eliminate these situations preemptively, since the pickup based on Fig. 1 is applied before suppression of pickup based on Fig. 2. The suppression rules in Fig. 3 protect desirable junction formations. The pickup triggers in Fig. 1 and pickup suppressors in Fig. 3 are mutually exclusive, so the junction configurations are not preempted by Fig. 1 rules. However, excess blocks around the junction are removed by Fig. 1 pickups. The heuristic underlying the selection of certain junctions is that each junction should be the confluence of three branches (not four), and the branches should be as well separated as possible around the junction point.

4. Rules for deposit

Deposit is considered only when agent i is carrying a block and the current location is empty.

The condition for depositing a block depends on both an agent's view and its recent experience as given by its direction and counter memories. Essentially, an agent carrying a block tries to locate a continuous line of blocks, moves along it until it reaches the end, and then deposits its block.

The rule for deposit is that: if $l_i(t)=0$ (cell has no block) and $c_i(t)>0$, agent i deposits its block.

The reason behind this rule will be clarified below as the rules for movement are considered.

5. Rules for movement

The rules for movement at t depend on whether the agent is carrying a block, i.e., $s_i(t)=1$.

Case I: Agent is free $[s_i(t)=0]$. In this case, the agent randomly chooses one of the eight possible directions and moves to the corresponding cell.

Case II: Agent is loaded $[s_i(t)=1]$.

(1) If current position is empty $[l_i(t)=0]$: The agent randomly chooses one of the eight possible directions and moves to the corresponding cell. Note that one of the cases where this condition occurs is when the agent picks up an item at $l_i(t)$ at time t.

(2) If current position is occupied $[l_i(t)=1]$. We have the following

(a) *If this is first encounter with a new block group* $[c_i(t)=0]$. We get the following facts

(i) If the block at $l_i(t)$ is isolated [i.e., $v^3_{i_k}(t)=0$ for $k=1,\ldots,8]$ (α) The agent randomly chooses one of the eight possible directions and moves to the corresponding cell, (β) $c_i(t+1)=c_i(t)=0$.

(ii) If the block at $l_i(t)$ is not isolated [i.e., $v^3_{i_k}(t)=1$ for some $k=1,\ldots,8]$ (α) the agent checks the neighbors of $l_i(t)$ in random order and moves to the first block-occupied cell it finds, (β) $c_i(t+1)=c_i(t)+1$.

In this case, the agent is moving along one block segment passing through $l_i(t)$.

(b) *If this is not the first encounter with a new block group* $[c_i(t)>0]$. We get the following facts

(i) The agent checks the neighbor of $l_i(t)$ along direction $d_i(t)$ and to either side of it. These are termed the *forward view cells.*

(ii) If any of the forward view cells are block occupied, the agent moves to one of these occupied cells at random.

(iii) If none of the forward view cells are block occupied, the agent moves to one of the three cells at random.

(iv) $c_i(t+1)=c_i(t)+1$.

After this move, if $l_i(t+1)$ is empty, the that indicates the agent is in a cell adjacent to the end of a block segment [since $c_i(t+1)>0$]. This triggers a deposit, extending the segment. And if $l_i(t+1)$ is block occupied, the agent has simply moved a step along the block segment being followed.

B. Stopping criteria

While the emergent structure develops rapidly during early stages of a simulation, change becomes very slow once the vast majority of blocks are in locally viable arrangements. In order to obtain data in a reasonable length of time, we set criteria to stop a simulation when it appears sufficiently close to structural convergence. The simulation stops if we have the following.

(1) No agent has picked up or deposited any items in the previous 500 iterations. This stopping criterion is used for the case when $\alpha=0$.

(2) More than 98% of the items are part of the largest cluster. Application of this stopping criterion involves periodic checks of $M(t)$, the size of the largest cluster. It is needed mainly when $\alpha>0$, i.e., terminal blocks are picked up with finite probability.

(3) The agents have reached a time limit. The time limit used during all the simulations is 1 000 000 time steps. It is needed for very low block densities when $\alpha>0$.

IV. RESULTS AND DISCUSSION

As indicated above, the structure obtained at the end of each simulation is analyzed in terms of clustering and, in particular, the size of the largest cluster. This corresponds to the usual method in cases of percolation, where the largest cluster or component is studied as a function of a parameter and system size [16]. In this paper, the parameter of interest is the block density $D=N/L^2$.

The behavior of the system is very different for the $\alpha=0$ and $\alpha>0$ cases, so we consider them separately.

A. Case I: $\alpha=0$

In this case, agents do not pick up terminal blocks in segments they encounter. Thus, short, isolated block segments are free to form and persist.

Figure 5 shows plots of the relative size $\sigma=M/N$, of the largest cluster against block density D for environments of size $L=100$ and $L=200$. In both cases, σ shows highly nonlinear behavior with respect to D. The most noticeable feature is an abrupt transition at a critical density $D_c\approx 0.235$, where σ increases from near 0 to near 1, indicating a phase transition [16]. The critical density in both cases is the same, indicating size invariance. Also, as implied by theory, the transition is tending to become steeper with increasing L. True criticality would, of course, be obtained only in the limit $L\rightarrow\infty$.

Figure 6 shows an example of the block structure formed when block density is just below the critical value. In this case, $D=0.2$ and $\sigma=0.10$. The largest cluster is indicated by the darker blocks, and is clearly much smaller than the size of the system. Figure 7, in contrast, shows a structure formed with block density just above the critical threshold at D

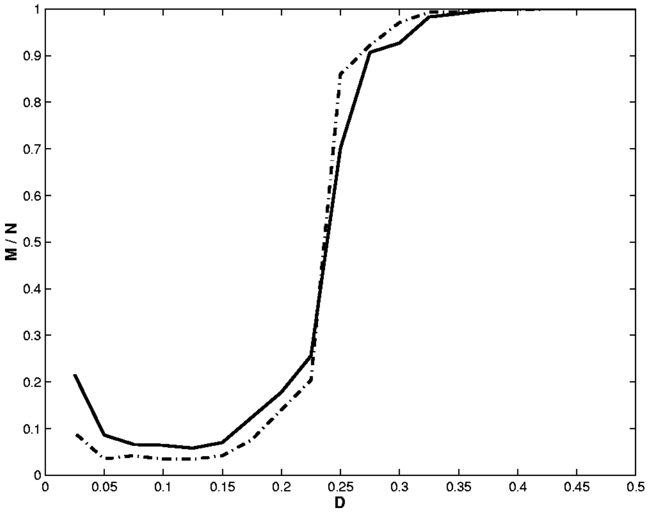

FIG. 5. Largest cluster size σ as a function of block density D for a 100×100 environment (solid line) and a 200×200 environment (dashed line).

$=0.27$. In this case, the empirical value of $\sigma=0.87$, indicating that 87% of all blocks are members of it.

As is well known from percolation theory [16], at the critical threshold, cluster size scales as a power law with the size of the environment, L. Figure 8 plots M as a function of L at $D=0.235$. The log-log plot shows excellent agreement with a power law relationship, $M\sim L^{\beta}$, with $\beta\approx1.95$. The value of β can be interpreted as a fractal dimension of the largest cluster [16], and the empirical value we obtain is in close agreement with those seen in well-known examples of percolation in two-dimensional square lattices [18].

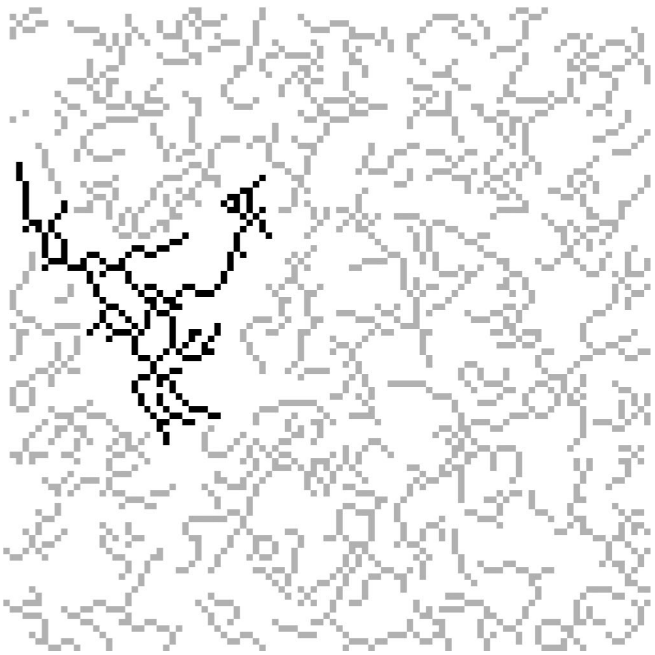

FIG. 6. Final block structure for a simulation with $D=0.2$ for a 100×100 environment. The darker blocks indicate the largest cluster.

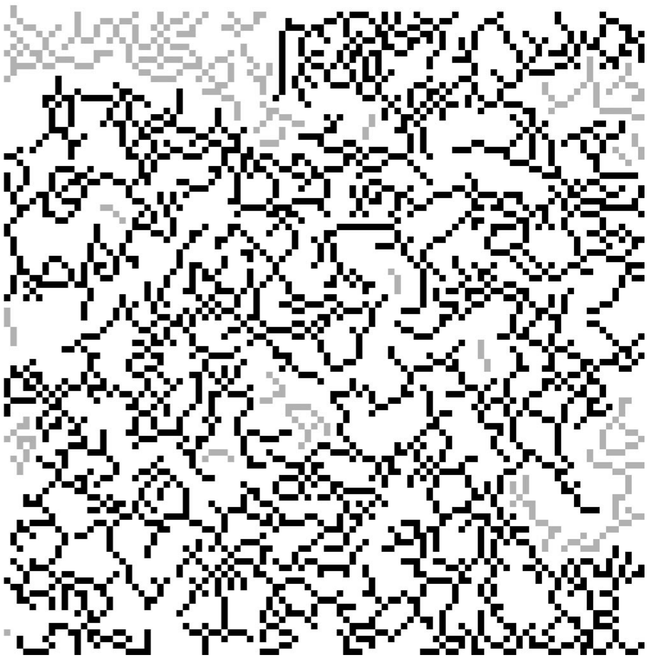

FIG. 7. Final block structure for a simulation with $D=0.27$ for a 100×100 environment. The darker blocks indicate the largest cluster.

Considering the results in Fig. 5 more closely, it is possible to distinguish three distinct regimes along the D axis:

Regime I. This regime occurs at extremely low block densities, and has a σ significantly higher than zero, though not near 1. This is a finite-size effect due to the fact that clusters can only coalesce at sites that have groups of two or more connected blocks in the initial distribution. We call these *seed sites*. At very low block densities, the initial distribution has only a very small number of seed sites that attract all the other isolated blocks. Since the absolute number of blocks is small, the largest cluster can easily acquire a significant *fraction* of blocks, leading to relatively large σ values, even

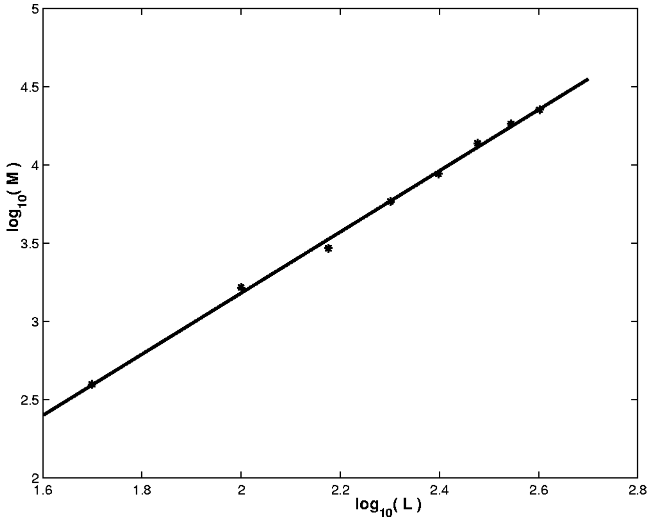

FIG. 8. Size M of largest cluster vs size L of the environment with $D=0.235$, the approximate critical density.

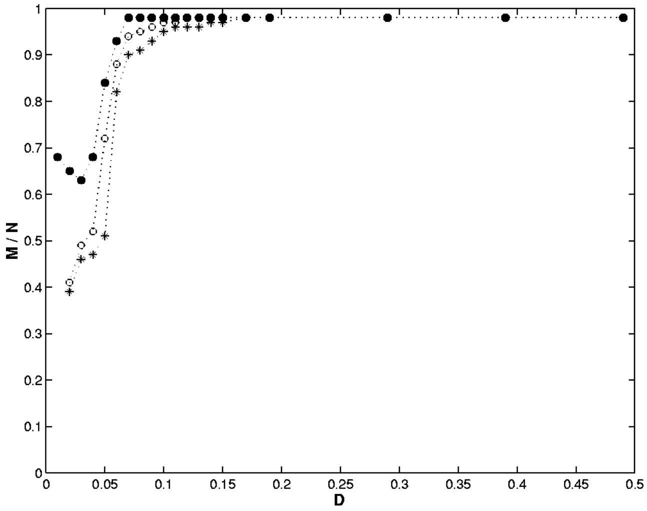

FIG. 9. Cluster size as a function of block density for $\alpha=0.1$ (full circles), $\alpha=0.5$ (empty circles), and $\alpha=0.9$ (asterisks).

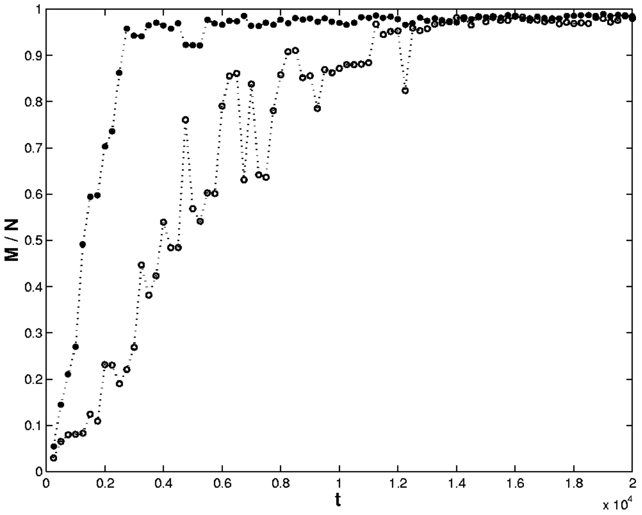

FIG. 10. Temporal evolution of σ when $\alpha=0.1$ and $D=0.15$ (empty circles) and 0.2 (filled circles).

though the absolute size of the cluster is very small. We expect this effect to disappear on much larger lattices, and indeed, Fig. 5 shows that it is weaker for the $L=200$ lattice than for the $L=100$ lattice.

Regime II. This is the classic subpercolation regime, where the largest cluster is of insignificant relative size. It occurs for low densities up to the percolation threshold. The block density in this range is sufficient to create a large number of cluster seed sites in the initial distribution, all of which compete for the relatively small number of blocks. As a result, none of them can acquire a significant fraction of blocks.

Regime III. This is the percolation regime, where the initial seed sites are dense enough and the number of blocks large enough so that initially disconnected clusters can connect via the actions of the agents to form one large cluster.

It should be noted that the rules governing pickup and depositing of blocks allow clusters to be broken up if they do not conform to certain characteristics (e.g., if a junction is not acceptable under the rules of Fig. 3). Thus, while clusters can only begin at a seed site, not all seed sites grow clusters—some just break up. There is a continuous process of construction and destruction until the vast majority of blocks are in acceptable positions. When block density is relatively high, more and more blocks get into acceptable positions, making the destruction process increasingly weak and eventually leading to (near)convergence (see Fig. 10). However, when the block density is very low, acceptable positions for blocks are also rare, loaded agents wander about for long periods looking for a deposit site, and the process does not really converge. For this reason, in Figs. 5 and 9 (below), we have plotted the largest cluster seen over the duration of the simulation rather than the size at the end. For all but the lowest block densities, this makes no difference, since the cluster size converges (see Fig. 10), but at very low D, the cluster size fluctuates significantly. Thus, data shown for these values form, at best, an upper bound on the mean size of the largest cluster.

B. Case II: $\alpha>0$

When agents are allowed to pick up terminal blocks, the system's behavior changes dramatically, so that the final structure comprises a single large cluster except at very low block densities. Essentially, allowing small, isolated segments to be broken up ensures that all blocks have a chance to move into the largest cluster.

Figure 9 shows the relative size of the largest cluster size as a function of D for $\alpha=0.1$, 0.5, and 0.9 in a 100×100 lattice. The subpercolation regime with very low cluster size has contracted dramatically and virtually disappeared. This is because the percolation threshold has become so low that, in a finite-sized lattice, the system goes from regime III (percolation) to I instead of regime II. Thus, the blocks are able to coalesce into a small number of clusters, and the largest of these typically comprises a significant fraction of blocks due to finite lattice size. As discussed earlier, clustering at very low D is quite unstable. For $D\leqslant0.1$, we ran simulations for 1×10^6 time steps each, but the cluster size shown for these values should be seen only as an upper bound. Also, for high α and $D\leqslant0.01$, we were not able to obtain any clustering at all because the seed sites were broken up before clustering could even begin. As the lattice size $L\rightarrow\infty$, we expect the σ for extremely low densities to approach zero. However, for all densities above this very low threshold, only one dominant cluster exists.

Figure 10 shows the temporal evolution of σ in a particular simulation with $\alpha=0.1$ and two values of D. Initially, the structure is clumped into small clusters, but gradually almost all blocks move into the dominant cluster, and cluster size converges. It is noticeable how σ does not increase monotonically, but undergoes occasional abrupt reductions reminiscent of an annealing process. These reductions are a signature of the destructive processes that occasionally break up clusters (as discussed above).

Finally, Fig. 11 shows the time taken for 95% of blocks in the $D=0.3$ case to join the largest cluster. This shows that it takes longer for all the blocks to end up in one single cluster

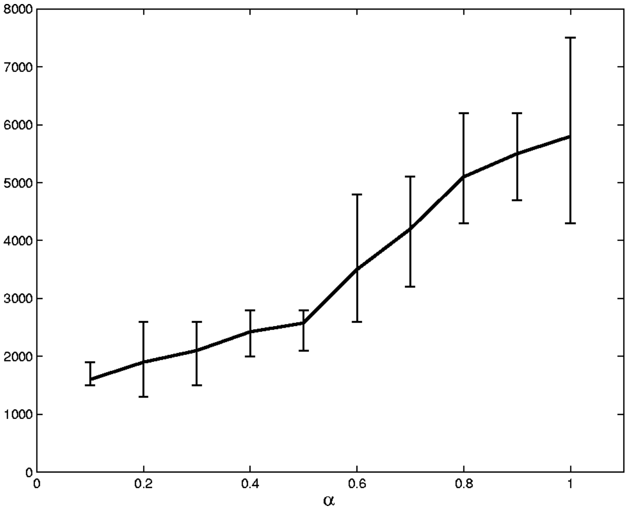

FIG. 11. Mean clustering time (95% clustering) when $D=0.3$.

when α increases. This may be due to the fact that at higher values of α, segment terminals are being broken up and rearranged too fast, thus reducing the chance that those segment terminals could end up connecting other clusters. However, Fig. 11 also shows that almost any probability of picking up terminal segments is sufficient to ensure an almost totally connected structure at this D.

C. Segment terminals

An interesting aspect of the emergent structure that is not captured by cluster size is the way the block segments are connected. We consider whether a large fraction of segments form “filaments” (linear structures with one unattached terminal) as opposed to “loops”—in other words, to what degree is the graph of the structure “treelike.” We do this by counting the number λ of unattached segment terminals in the final structure. Figure 12 shows how λ/N varies as α changes for two values of D. When $\alpha=0$, the relative count of segment terminals is high, but it decreases quickly as α increases. This may be explained by the fact that at high values of α, the segment terminals are being broken up and rearranged a lot more than at lower values of α. This increases the chance that those segment terminals will end up forming loops, creating graphs with multiple paths between locations.

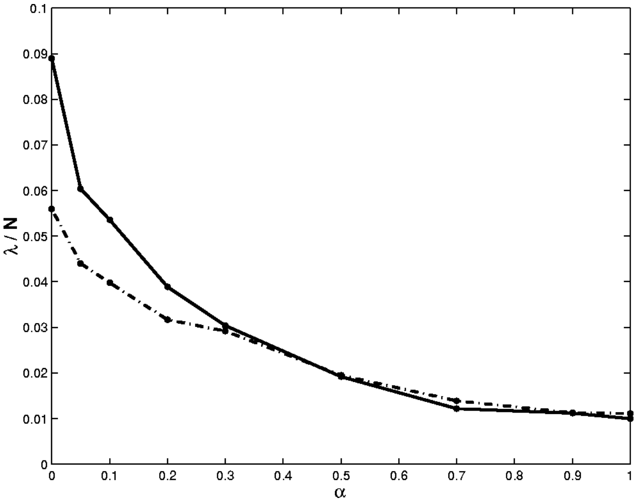

FIG. 12. Number of segment terminals, λ, v/s α for a $L=100$ lattice when $D=0.3$ (solid line) and 0.35 (dashed line).

V. CONCLUSION

In this paper, we have investigated the clustering properties of structures formed by simple agents picking up and depositing blocks in a lattice environment under simple, local rules. Our empirical findings indicate the occurrence of a phase transition in the self-organized structure. However, this phase transition changes dramatically if agents are allowed to preferentially destroy short, isolated block segments. This ensures that all blocks become part of a single connected structure except at very low densities . It should be noted that seeking—or even detecting—such global connectivity is not part of the agents' behavioral rules, and no agent is aware of its existence. It is entirely an emergent phenomenon, arising out of the application of local rules through a process of self-organization. We expect that the results presented here will be useful in the design and analysis of swarm-based methods for self-organized construction. The ability to obtain fully connected structures may well be the first requirement for most other, more detailed construction algorithms, since connected structures can propagate information over long distances via local communication. Such information propagation will be essential in order to obtain more controlled and regular structures such as those studied in amorphous computing [21–24].

[1] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems* (MIT Press, Cambridge, MA, 1999).

[2] N.R. Franks, A. Wilby, V.W. Silverman, and C. Tofts, Anim. Behav. **44**, 357 (1992).

[3] G. Theraulaz and E. Bonabeau, Science **269**, 686 (1995).

[4] G. Theraulaz and E. Bonabeau, J. Theor. Biol. **177**, 381 (1995).

[5] N.R. Franks and J.-L. Deneubourg, Anim. Behav. **54**, 779 (1997).

[6] E. Bonabeau *et al.*, Philos. Trans. R. Soc. London, Ser. B **353**, 1561 (1998).

[7] J.-L. Deneubourg, S. Goss, N.R. Franks, and J.M. Pasteels, J. Insect Behav. **2**, 719 (1989).

[8] N.R. Franks, N. Gomez, S. Goss, and J.-L. Deneubourg, J. Insect Behav. **4**, 583 (1991).

[9] E. Bonabeau, G. Theruaulaz, and J.-L. Deneubourg, Proc. R. Soc. London, Ser. B **263**, 1565 (1996).

[10] E. Bonabeau, G. Theruaulaz, and J.-L. Deneubourg, Bull. Math. Biol. **60**, 753 (1998).

[11] R.C. Kube and H. Zhang, Adapt. Behav. **2**, 189 (1993).

[12] M. Mataric, Rob. Auton. Syst. **16**, 321 (1995).

[13] Y. Cao, A.S. Fukunaga, and A.B. Kahng, Auton. Rob. **4**, 7 (1997).

[14] R. C. Arkin, *Behavior-Based Robotics* (MIT Press, Cambridge, MA, 1998).

[15] E.M. Rauch, M.M. Millonas, and D.R. Chailvo, Phys. Lett. A **207**, 185 (1995).

[16] D. Stauffer and A. Aharony, *Percolation Theory* (Taylor & Francis, London, 1992).

[17] J. Feder, *Fractals* (Plenum, New York, 1988).

[18] H.-O. Peitgen, H. Jürgens, and D. Saupe, *Chaos and Fractals: New Frontiers of Science* (Springer-Verlag, New York, 1992).

[19] R. Albert and A.-L. Barabási, Rev. Mod. Phys. **74**, 47 (2002).

[20] J. M. Garcia-Ruiz, E. Louis, P. Meakin, and L. M. Sander, *Growth Patterns in Physical Sciences and Biology* (Plenum Press, New York, 1993).

[21] H. Abelson *et al.*, Commun. ACM **43**, 74 (2000).

[22] D. Coore, Ph.D. thesis, MIT AI Lab, 1999 (unpublished).

[23] R. Nagpal, Ph.D. thesis, MIT AI Lab, 2001 (unpublished).

[24] R. Nagpal, Proceedings of the Fourth International Conference on Complex Systems, 2002 (unpublished).

[25] K. Sugihara and I. Suzuki, J. Rob. Syst. **13**, 127 (1996).

[26] T. Balch and R.C. Arkin, IEEE Trans. Rob. Autom. **14**, 926 (1998).

[27] J. Fredslund and M.J. Matarić, IEEE Trans. Rob. Autom. **18**, 837 (2002).