A Stochastic Approach to Shortcut Bridging in Programmable Matter

Marta Andrés Arroyo[1], Sarah Cannon[2⋆], Joshua J. Daymude[3⋆⋆], Dana Randall[2†], and Andréa W. Richa[3‡]

[1] University of Granada, Spain, martaandres@correo.ugr.es
[2] College of Computing, Georgia Institute of Technology, USA, sarah.cannon@gatech.edu, randall@cc.gatech.edu
[3] Computer Science, CIDSE, Arizona State University, USA, {jdaymude,aricha}@asu.edu

Abstract. In a *self-organizing particle system*, an abstraction of programmable matter, simple computational elements called *particles* with limited memory and communication self-organize to solve system-wide problems of movement, coordination, and configuration. In this paper, we consider stochastic, distributed, local, asynchronous algorithms for "shortcut bridging," in which particles self-assemble bridges over gaps that simultaneously balance minimizing the length and cost of the bridge. Army ants of the genus *Eticon* have been observed exhibiting a similar behavior in their foraging trails, dynamically adjusting their bridges to satisfy an efficiency tradeoff using local interactions [1]. Using techniques from Markov chain analysis, we rigorously analyze our algorithm, show it achieves a near-optimal balance between the competing factors of path length and bridge cost, and prove that it exhibits a dependence on the angle of the gap being "shortcut" similar to that of the ant bridges. We also present simulation results that qualitatively compare our algorithm with the army ant bridging behavior. The proposed algorithm demonstrates the robustness of the stochastic approach to algorithms for programmable matter, as it is a surprisingly simple generalization of a stochastic algorithm for compression [2].

1 Introduction

In developing a system of *programmable matter*, one endeavors to create a material or substance that can utilize user input or stimuli from its environment to change its physical properties in a programmable fashion. Many such systems have been proposed (e.g., DNA tiles, synthetic cells, and reconfigurable modular

⋆ Supported in part by NSF DGE-1148903 and a grant from the Simons Foundation (#361047 to Sarah Cannon).
⋆⋆ Supported in part by NSF CCF-1637393.
† Supported in part by NSF CCF-1637031 and CCF-1526900.
‡ Supported in part by NSF CCF-1637393 and CCF-1422603.

robots) and each attempts to perform tasks subject to domain-specific capabilities and constraints. In our work on *self-organizing particle systems*, we abstract away from specific settings and envision programmable matter as a system of computationally limited devices (which we call *particles*) which can actively move and individually execute distributed, local, asynchronous algorithms to cooperatively achieve macro-scale tasks of movement and coordination.

The phenomenon of local interactions yielding emergent, collective behavior is often found in natural systems; for example, honey bees choose hive locations based on decentralized recruitment [3] and cockroach larvae perform self-organizing aggregation using pheromones with limited range [4]. In this paper, we present an algorithm inspired by the work of Reid et al. [1], who found that army ants continuously modify the shape and position of foraging bridges — constructed and maintained by their own bodies — across holes and uneven surfaces in the forest floor. Moreover, these bridges appear to stabilize in a structural formation which balances the "benefit of increased foraging trail efficiency" with the "cost of removing workers from the foraging pool to form the structure" [1].

We attempt to capture this inherent trade-off in the design of our algorithm for "shortcut bridging" in self-organizing particle systems (to be formally defined in Section 1.3). Our proposed algorithm for shortcut bridging is an extension of the stochastic, distributed algorithm for the *compression problem* introduced in [2], which shows that many fundamental elements of our stochastic approach can be generalized to applications beyond the specific context of compression. In particular, our stochastic approach may be of future interest in the molecular programming domain, where simpler variations of bridging have been studied. Groundbreaking works in this area, such as that of Mohammed et al. [5], focus on forming molecular structures that connect some fixed points; our work may offer insights on further optimizing the quality and/or cost of the resulting bridges.

Shortcut bridging is an attractive goal for programmable matter systems, as many application domains envision deploying programmable matter on surfaces with structural irregularities or dynamic topologies. For example, one commonly imagined application of smart sensor networks is to detect and span small cracks in infrastructure such as roads or bridges as they form; dynamic bridging behavior would enable the system to remain connected as the cracks form and to shift its position accordingly.

1.1 Related Work

When examining the recently proposed and realized systems of programmable matter, one can distinguish between *passive* and *active* systems. In passive systems, computational units cannot control their movement and have (at most) very limited computational abilities, relying instead on their physical structure and interactions with the environment to achieve locomotion (e.g., [6–8]). A large body of research in molecular self-assembly falls under this category, which has mainly focused on shape formation (e.g., [9–11]). Rather than focusing on constructing a specific fixed target shape, our work examines building dynamic bridges whose exact shape is not predetermined. Mohammed et al. studied the

more relevant problem of connecting DNA origami landmarks with DNA nanotubes, using a carefully designed process of nanotube nucleation, growth, and diffusion to achieve and maintain the desired connections [5]. The most significant differences between their approach and ours is (*i*) the bridges we consider already connect their endpoints at the start, and focus on the more specific goal of optimizing their shape with respect to a parameterized objective function, and (*ii*) our system is active as opposed to passive.

Active systems, in contrast, are composed of computational units which can control their actions to solve a specific task. Examples include *swarm robotics*, various other models of modular robotics, and the *amoebot model*, which defines our computational framework (detailed in Section 1.2).

Swarm robotics systems usually involve a collection of autonomous robots that move freely in space with limited sensing and communication ranges. These systems can perform a variety of tasks including gathering [12], shape formation [13, 14], and imitating the collective behavior of natural systems [15]; however, the individual robots have more powerful communication and processing capabilities than those we consider. *Modular self-reconfigurable robotic systems* focus on the motion planning and control of kinematic robots to achieve dynamic morphology [16], and *metamorphic robots* form a subclass of self-reconfiguring robots [17] that share some characteristics with our geometric amoebot model. Walter et al. have conducted some algorithmic research on these systems (e.g., [18, 19]), but focus on problems disjoint from those we consider.

In the context of molecular programming, our model most closely relates to the *nubot* model by Woods et al. [20, 21], which seeks to provide a framework for rigorous algorithmic research on self-assembly systems composed of active molecular components, emphasizing the interactions between molecular structure and active dynamics. This model shares many characteristics of our amoebot model (e.g., space is modeled as a triangular grid, nubot monomers have limited computational abilities, and there is no global orientation) but differs in that nubot monomers can replicate or die and can perform coordinated rigid body movements. These additional capabilities prohibit the direct translation of results under the nubot model to our amoebot model.

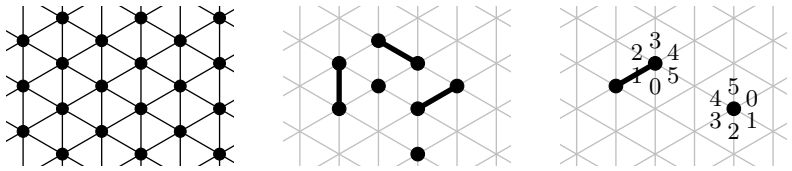
1.2 The Amoebot Model

We recall the main properties of the *amoebot model* [2, 22], an abstract model for programmable matter that provides a framework for rigorous algorithmic research on nano-scale systems. We represent programmable matter as a collection of individual computational units known as *particles*. The structure of a particle system is represented as a subgraph of the infinite, undirected graph $G = (V, E)$, where V is the set of all possible locations a particle could occupy and E is the set of all possible atomic transitions between locations in V. For shortcut bridging (and many other problems), we assume the *geometric* amoebot model, in which $G = \Gamma$, the *triangular lattice* (Figure 1a).

Each particle is either *contracted*, occupying a single location, or *expanded*, occupying or a pair of adjacent locations in Γ (Figure 1b). Particles move via

a series of *expansions* and *contractions*; in particular, a contracted particle may expand into an adjacent unoccupied location, and completes its movement by contracting to once again occupy a single location.

Two particles occupying adjacent locations in Γ are said to be *neighbors*. Each particle is *anonymous*, lacking a unique identifier, but can locally identify each of its neighbors via a collection of ports corresponding to edges incident to its location. We assume particles have a common *chirality*, meaning they share the same notion of *clockwise direction*, which allows them to label their ports in clockwise order. However, particles do not share a global orientation and thus may have different offsets for their port labels (Figure 1c).



(a) (b) (c)

Fig. 1: (a) A section of the triangular lattice Γ; (b) expanded and contracted particles; (c) two non-neighboring particles with different offsets for their port labels.

Each particle has a constant-size, local memory that can read from for communication by it and its neighbors, so a particle's state (e.g., contracted or expanded) is visible to its neighbors. Due to the limitation of constant-size memory, a particle cannot know the total number of particles in the system or any estimate of it. We assume the standard asynchronous model from distributed computing [23], where progress is achieved through *atomic particle activations*. Once activated, a particle can perform an arbitrary, bounded amount of computation involving its local memory and the memories of its neighbors, and can perform at most one movement. A classical result under this model states that for any concurrent asynchronous execution of activations, there is a sequential ordering of activations producing the same result, provided conflicts that arise in the concurrent execution are resolved. In our scenario, conflicts arising from simultaneous memory writes or particle expansions into the same empty location are assumed to be resolved arbitrarily. Thus, while many particles may be activated at once in a realistic settings, it suffices to consider a sequence of activations in which only one particle is active at a time.

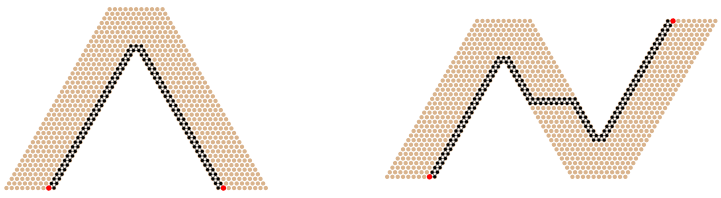
1.3 Problem Description

Just as the uneven surfaces of the forest floor affect the foraging behavior of army ants, the collective behavior of particle systems should change when Γ is non-uniform. Here, we focus on system behaviors when the vertices of Γ are either *gap* (unsupported) or *land* (supported) locations. A particle occupying some

location in Γ can tell whether it is in the gap or on land. We also introduce *objects*, which are static particles that do not perform computation; these are used to constrain the particles to remain connected to certain fixed sites. In order to analyze the strength of the solutions our algorithm produces, we define the *weighted perimeter* $\overline{p}(\sigma, c)$ of a particle system configuration σ to be the summed edge weights of edges on the external boundary of σ, where edges on land have weight 1, edges in the gap have a fixed weight $c > 1$, and edges with one endpoint on land and one endpoint in the gap have weight $(1 + c)/2$.

In the *shortcut bridging problem*, we consider an instance $(L, O, \sigma_0, c, \alpha)$ where $L \subseteq V$ is the set of land locations, O is the set of (two) objects to be bridged between, σ_0 is the initial configuration of the particle system, $c > 1$ is a fixed weight for gap edges, and $\alpha > 1$ is the accuracy required of a solution. An instance is valid if (i) the objects of O and particles of σ_0 all occupy locations in L, (ii) σ_0 connects the objects, and (iii) σ_0 is connected, a notion formally defined in Section 2.1. An algorithm solves an instance if, beginning from σ_0, it reaches and remains (with high probability) in a set of configurations Σ^* such that any $\sigma \in \Sigma^*$ has perimeter weight $\overline{p}(\sigma, c)$ within an α-factor of its minimum value. The weighted perimeter balances in one function (using an appropriate weight for the land and gap perimeter edges) the trade-off observed in [1] between the competing objectives of establishing a short path between the fixed endpoints while not having too many particles in the gap. We focus on gap perimeter instead of the number of particles in the gap (which is perhaps a more natural analogy to [1]) because (1) the shortcut bridges produced with this metric more closely resemble the ant structures and (2) only particles on the perimeter of a configuration can move, and thus recognize the potential risk of being in the gap, justifying our focus on perimeter in the weight function.

In analogy to the apparatus used in [1] (see Figure 3a), we are particularly interested in the special case where L forms a V-shape, O has two objects positioned at either base of L, and σ_0 lines the interior sides of L, as in Figure 2a. However, our algorithm is not limited to this setting; for example, we show simulation results for an N-shaped land mass (Figure 2b) in Section 5.



(a) (b)

Fig. 2: Examples of L, O and σ_0 for instances of the shortcut bridging problem for which we present simulation results (Section 5). Light (brown) nodes are land locations, large (red) nodes are occupied by objects, and black nodes are occupied by particles.

1.4 The Stochastic Approach to Self-Organizing Particle Systems

In [2], we introduced a stochastic, distributed algorithm for compression in the amoebot model; here we extend that work to show that stochastic approach is in fact more generally applicable. The motivation underlying this Markov chain approach to programmable matter comes from statistical physics, where ensembles of particles reminiscent of the amoebot model are used to study physical systems and demonstrate that local micro-behavior can induce global macro-scale changes to the system [24–26]. Like a spring relaxing, physical systems favor configurations that minimize energy. The energy function is determined by a *Hamiltonian* $H(\sigma)$. Each configuration σ has weight $w(\sigma) = e^{-B \cdot H(\sigma)}/Z$, where where $B = 1/T$ is inverse temperature and $Z = \sum_\tau e^{-B \cdot H(\tau)}$ is the normalizing constant known as the *partition function*.

For shortcut bridging, we introduce a Hamiltonian over particle system configurations so that the configurations of interest will have the lowest energy, and will design our algorithms to favor these low energy configurations. We assign each particle system configuration σ a Hamiltonian $H(\sigma) = \overline{p}(\sigma, c)$, its weighted perimeter. Setting $\lambda = e^B$, we get $w(\sigma, c) = \lambda^{-\overline{p}(\sigma,c)}/Z$. As λ gets larger (by increasing B, effectively lowering temperature), we increasingly favor configurations where $\overline{p}(\sigma, c)$ is small and the desired bridging behavior is exhibited. We prove (Theorem 1) that raising λ above $2 + \sqrt{2}$ suffices for the low energy configurations with small $\overline{p}(\sigma, c)$ to dominate the state space and overcome the entropy of the system. That is, for $\lambda > 2 + \sqrt{2}$, low energy configurations occur with sufficient frequency that we will find such configurations when we sample over the whole state space. The key tool used to establish this is a careful *Peierls argument*, used in statistical physics to study non-uniqueness of limiting Gibbs measures and to determine the presence of phase transitions and in computer science to establish slow mixing of Markov chains (see, e.g., [27], Chapter 15).

Compared to other algorithms for programmable matter and self-organizing particle systems, stochastic methods such as the compression algorithm of [2] and our shortcut bridging algorithm are nearly oblivious, more robust to failures, and require little to no communication between particles. Because each of these algorithms is derived from a stochastic process, powerful tools developed for Markov chain analysis can be employed to rigorously understand their behavior.

1.5 A Stochastic Algorithm for Shortcut Bridging

We present a Markov chain $\mathcal{M}$ for *shortcut bridging* in the geometric amoebot model which translates directly to a fully distributed, local, asynchronous algorithm $\mathcal{A}$. We prove that $\mathcal{M}$ (and by extension, $\mathcal{A}$) solves the shortcut bridging problem: for any constant $\alpha > 1$, the long run probability that $\mathcal{M}$ is in a configuration σ with $\overline{p}(\sigma, c)$ larger than α times its minimum possible value is exponentially small. We then specifically consider V-shaped land masses with an object on each branch of the V, and prove that the resulting bridge structures vary with the interior angle of the V-shaped gap being shortcut — a phenomenon also observed by Reid et al. [1] in the army ant bridges — and show in simulation that they are qualitatively similar to those of the ants (e.g., Figure 3).

(a) (b)

Fig. 3: (a) In this image from [1], army ants of the genus *Eticon* build a dynamic bridge which balances the benefit of a shortcut path with the cost of committing ants to the structure. (b) Our shortcut bridging algorithm also balances competing objectives and converges to similar configurations.

2 Background

2.1 Terminology for Particle Systems

For a particle P (resp., location ℓ), we use $N(P)$ (resp., $N(\ell)$) to denote the set of particles and objects[1] adjacent to P (resp., to ℓ). For adjacent locations ℓ and ℓ', we use $N(\ell \cup \ell')$ to denote the set $N(\ell) \cup N(\ell')$, not including particles or objects occupying either ℓ or ℓ'.

We define an *edge* of a particle configuration to be an edge of Γ where both endpoints are occupied by particles. When referring to a *path*, we mean a path in the subgraph of Γ induced by the locations occupied by particles. Two particles are *connected* if there exists a path between them, and a configuration is *connected* if all pairs of particles are. A *hole* in a configuration is a maximal finite component of adjacent unoccupied locations. We specifically consider connected configurations with no holes, and our algorithm, if starting at such a configuration, will maintain these properties.

Let σ be a connected configuration with no holes. The *perimeter* of σ, denoted $p(\sigma)$, is the length of the walk around the (single external) boundary of the particles. The *gap perimeter* of σ, denoted $g(\sigma)$, is the number of perimeter edges that are in the gap, where edges with one endpoint in the gap and one endpoint on land count as half an edge in the gap. Note that an edge may appear twice in the boundary walk, and thus may be counted twice in $p(\sigma)$ or $g(\sigma)$.

2.2 Markov Chains

Our distributed shortcut bridging algorithm is based on a Markov chain, so we briefly review the necessary terminology. A *Markov chain* is a memoryless

[1] The notion of a particle (resp., location) neighborhood $N(P)$ (resp., $N(\ell)$) has been extended from [2] to include objects.

stochastic process defined on a finite set of states Ω. The transition matrix P on $\Omega \times \Omega \to [0,1]$ is defined so that $P(x,y)$ is the probability of moving from state x to state y in one step, for any pair of states $x, y \in \Omega$. The t-step transition probability $P^t(x,y)$ is the probability of moving from x to y in exactly t steps.

A Markov chain is *ergodic* if it is *irreducible*, i.e., for all $x, y \in \Omega$, there is a t such that $P^t(x,y) > 0$, and *aperiodic*, i.e., for all $x, y \in \Omega$, g.c.d. $\{t : P^t(x,y) > 0\} = 1$. Any finite, ergodic Markov chain converges to a unique *stationary distribution* π given by, for all $x, y \in \Omega$, $\lim_{t\to\infty} P^t(x,y) = \pi(y)$. Any distribution π' satisfying $\pi'(x)P(x,y) = \pi'(y)P(y,x)$, for all $x, y \in \Omega$, (the *detailed balance condition*) and $\sum_{x\in\Omega} \pi'(x) = 1$ is the unique stationary distribution of the Markov chain (see, e.g., [28]).

Given a Markov chain and a desired stationary distribution π on Ω, the celebrated Metropolis-Hastings algorithm [29] defines appropriate transition probabilities for the chain so that π is its stationary distribution. Starting at $x \in \Omega$, pick a neighbor y in Ω uniformly with some fixed probability (that is the same for all x), and move to y with probability $\min\{1, \pi(y)/\pi(x)\}$; with the remaining probability stay at x and repeat. Using detailed balance, if the state space is connected then π must be the stationary distribution. While calculating $\pi(x)/\pi(y)$ seems to require global knowledge, this ratio can often be calculated using only local information when many terms cancel out. In our case, the Metropolis probabilities are simply $\min\{1, \lambda^{\overline{p}(x,c)-\overline{p}(y,c)}\}$; if x and y only differ by one particle P, as is the case with all moves of our algorithm, then $\overline{p}(x,c) - \overline{p}(y,c)$ can be calculated using only local information from the neighborhood of P.

3 A Stochastic Algorithm for Shortcut Bridging

Recall that for the shortcut bridging problem, we desire for our algorithm to achieve small weighted perimeter, where boundary edges in the gap cost more than those on land. The algorithm must balance the competing objectives of having a short path between the two objects while not forming too large of a bridge. We capture these two factors by preferring small perimeter and small gap perimeter, respectively. While these objectives may appear to be aligned rather than competing, decreasing the length of the overall perimeter increases the gap perimeter and vice versa in the problem instances we consider (e.g., Figure 2).

Specifically, our Markov chain algorithm incorporates two bias parameters: λ and γ. The value of λ controls the preference for having a small perimeter, while γ controls the preference for having a small gap perimeter. In this paper, we only consider $\lambda > 1$ and $\gamma > 1$, which correspond to favoring a smaller perimeter and a smaller gap perimeter, respectively. Using a Metropolis filter, we ensure that our algorithm converges to a distribution over particle system configurations where the relative likelihood of the particle system being in configuration σ is $\lambda^{-p(\sigma)}\gamma^{-g(\sigma)}$, or equivalently, $\lambda^{-\overline{p}(\sigma,c)}$ for $c = 1 + \log_\lambda \gamma$. We note λ is the same parameter that controlled compression in [2], where particle configurations converged to a distribution proportional to $\lambda^{-p(\sigma)}$. That work showed that $\lambda > 1$

is not sufficient for compression to occur, so we restrict our attention to $\lambda > 2+\sqrt{2}$, the regime where compression provably happens.

To ensure that during the execution of our algorithm the particles remain connected and hole-free, we introduce two properties every movement must satisfy. These properties help to guarantee the local connectivity structure in the neighborhood of a moving particle doesn't change; more details may be found in [2]. Importantly, these properties maintain system connectivity[2], prevent holes from forming, and ensure reversibility of the Markov chain. These last two conditions are necessary for applying established tools from Markov chain analysis. Let ℓ and ℓ' be adjacent locations in Γ, and let $\mathbb{S} = N(\ell) \cap N(\ell')$ be the particles adjacent to both; we note $|\mathbb{S}| = 0, 1$, or 2.

Property 1. $|\mathbb{S}| \in \{1, 2\}$ and every particle in $N(\ell \cup \ell')$ is connected to a particle in $\mathbb{S}$ by a path through $N(\ell \cup \ell')$.

Property 2. $|\mathbb{S}| = 0$; ℓ and ℓ' each have at least one neighbor; all particles in $N(\ell) \setminus \{\ell'\}$ are connected by paths within this set; and all particles in $N(\ell') \setminus \{\ell\}$ are connected by paths within this set.

Importantly, these properties are symmetric with respect to ℓ and ℓ' and can be locally checkable by an expanded particle occupying both ℓ and ℓ' (as in Lines 2–3 of the Markov chain process described below).

We can now introduce the Markov chain $\mathcal{M}$ for an instance $(L, O, \sigma_0, c, \alpha)$ of shortcut bridging. For input parameter $\lambda > 2+\sqrt{2}$, set $\gamma = \lambda^{c-1}$, and beginning at initial configuration σ_0, which we assume has no holes,[3] repeat:

1. Select a particle P uniformly at random from among all n particles; let ℓ denote its location. Choose a neighboring location ℓ' and $q \in (0, 1)$ uniformly. Let σ be the configuration with P at ℓ and σ' the configuration with P at ℓ'.
2. If ℓ' is unoccupied, then P expands to occupy both ℓ and ℓ'. Otherwise, return to step 1.
3. If (i) $|N(\ell)| \neq 5$, (ii) ℓ and ℓ' satisfy Property 1 or Property 2, and (iii) $q < \lambda^{p(\sigma)-p(\sigma')}\gamma^{g(\sigma)-g(\sigma')}$, then P contracts to ℓ'. Otherwise, P contracts back to ℓ.

Although $p(\sigma) - p(\sigma')$ and $g(\sigma) - g(\sigma')$ are values defined at system-level scale, we show these differences can be calculated locally.

Lemma 1. *The values of $p(\sigma) - p(\sigma')$ and $g(\sigma) - g(\sigma')$ in Step 3(iii) of $\mathcal{M}$ can be calculated using information involving only ℓ, ℓ', and $N(\ell \cup \ell')$.*

Proof. These values only need to be calculated if 3(i) and 3(ii) are both true. By a result of [2], $p(\sigma) - p(\sigma') = |N(\ell')| - |N(\ell)|$, which is computable with only local information.

[2] Since particles treat objects as static particles, the particle system may actually disconnect into several components which remain connected through objects.

[3] If σ_0 has holes, our algorithm will eliminate them and they will not reform [2]; for simplicity, we focus only on the behavior of the system after this occurs.

Note $g(\sigma)$ is also the number of particles that are on the perimeter and in the gap, counted with appropriate multiplicity if a particle is on the perimeter more that once. Given a particle Q, let $G(Q)$ be 1 if Q is in a gap location and 0 if land; define $G(\ell)$ for a location ℓ similarly. Let $\delta(Q, \sigma)$ be the number of times Q appears on the perimeter of σ. Then $g(\sigma) = \sum_{Q \in p(\sigma)} G(Q)\delta(Q, \sigma)$. Define $\Delta(Q) = \delta(Q, \sigma) - \delta(Q, \sigma')$. For particles not in $\{P\} \cup N(\ell \cup \ell')$, $\Delta(Q) = 0$ as the neighborhood of Q will be identical in σ and σ'. Because steps $3(i)$ and $3(ii)$ are true, inspection shows this implies $\Delta(P) = 0$. Then:

$$g(\sigma) - g(\sigma') = \sum_{Q \in N(\ell \cup \ell')} G(Q)\Delta(Q) + \delta(P, \sigma)(G(\ell) - G(\ell')).$$

The second term above is calculable with only local information; for $Q \in N(\ell \cup \ell')$, to find $\Delta(Q)$ only Q's neighbors in this set need to be considered. If Q is adjacent to ℓ and not ℓ', $\Delta(Q) = -1$ if it has two neighbors in $N(\ell)$, $\Delta(Q) = 1$ if it has no neighbors in $N(\ell)$, and $\Delta(Q) = 0$ otherwise. If Q is adjacent to ℓ' but not ℓ, the opposite is true. If Q is adjacent to ℓ and ℓ', then $\Delta(Q) = 0$ if Q has zero or two neighbors in $N(\ell \cup \ell')$; $\Delta(Q) = 1$ if Q has a common neighbor with ℓ' but not ℓ; and $\Delta(Q) = -1$ if Q has a common neighbor with ℓ but not ℓ'. In all cases $\Delta(Q)$, and thus $g(\sigma) - g(\sigma')$, can be found with only local information. □

The state space Ω of $\mathcal{M}$ is the set of all configurations reachable from σ_0 via valid transitions of $\mathcal{M}$. We conjecture this includes all connected, hole-free configurations of n particles connected to both objects, but proving all such configurations are reachable from σ_0 is not necessary for our results. (The proof of the corresponding result in [2] does not generalize due to the presence of objects).

3.1 From $\mathcal{M}$ to a Distributed, Local Algorithm

While $\mathcal{M}$ is a Markov chain with centralized control of the particle system, one can transform $\mathcal{M}$ into a distributed, local, asynchronous algorithm $\mathcal{A}$ that each particle runs individually. The full details of this construction are given in [2], and we give a high level description here. When a particle is activated, it randomly chooses one of its six neighboring locations, checks if moving there is valid, and locally determines how the move will affect the global weight function $\lambda^{-p(\sigma)}\gamma^{-g(\sigma)}$. If the weight will increase, the particle performs the move; otherwise the particle only moves with some probability less than 1.

Specifically, in Step 1 of $\mathcal{M}$, a particle is chosen uniformly at random to be activated; to mimic this random activation sequence in a local way, we assume each particle has its own Poisson clock with mean 1 and becomes active after some random delay drawn from e^{-t}. During its activation, a contracted particle P occupying location ℓ chooses a neighboring location ℓ' and a real value $q \in (0, 1)$ uniformly at random[4], expanding into ℓ' if it is unoccupied, just as

[4] Note only a constant number of bits are needed to produce q, as λ and γ are constants and a particle move changes perimeter and gap perimeter by at most a constant.

in $\mathcal{M}$. However, unlike in $\mathcal{M}$, the expansion and contraction movements of P are necessarily split into two activations, since in the amoebot model a central assumption is that a particle can perform at most one movement per activation (see Section 1.2). Since P's two activations are not necessarily consecutive, P must be able to resolve conflicts with any other particles that may expand into its neighborhood before it becomes activated again and contracts. We accomplish this by introducing a system of Boolean flags maintained by all expanded particles. If P is the only expanded particle in its neighborhood, it stores a boolean flag $f = TRUE$ in its memory; otherwise, it sets $f = FALSE$. When P is activated again (now occupying both ℓ and ℓ'), it checks its flag f. If it is $FALSE$, P contracts back to ℓ, since some other particle in its neighborhood activated and expanded earlier. Otherwise, if f is $TRUE$, P checks the conditions in Step 3 of $\mathcal{M}$ and contracts either to ℓ or ℓ' accordingly. This ensures that at most one particle in a local neighborhood is moving at a time, mimicking the sequential nature of particle moves during the execution of Markov chain $\mathcal{M}$.

While this shows our Markov chain $\mathcal{M}$ can be translated into a fully local distributed algorithm with the same behavior, such an implementation is not always possible in general. Any Markov chain for particle systems that inherently relies on non-local moves of particles or has transition probabilities relying on non-local information cannot be executed by a local, distributed algorithm. Additionally, most distributed algorithms for amoebot systems are not stochastic; see, e.g., the mostly deterministic algorithms in [22, 30].

3.2 Properties of Markov Chain $\mathcal{M}$

We now show some useful properties of $\mathcal{M}$. Our first two claims follow from work in [2] and basic properties of Markov chains and our particle systems.

Lemma 2. *If σ_0 is connected and has no holes, then at every iteration of $\mathcal{M}$, the current configuration is connected and has no holes.*

Lemma 3. *$\mathcal{M}$ is ergodic.*

As $\mathcal{M}$ is finite and ergodic, it converges to a unique stationary distribution, and we can find that distribution using detailed balance.

Lemma 4. *The stationary distribution of $\mathcal{M}$ is given by*

$$\pi(\sigma) = \lambda^{-p(\sigma)}\gamma^{-g(\sigma)}/Z,$$

where $Z = \sum_{\sigma \in \Omega} \lambda^{-p(\sigma)}\gamma^{-g(\sigma)}$.

Proof. Properties 1 and 2 ensure that particle P moving from location ℓ to location ℓ' is valid if and only if P moving from ℓ' to ℓ is. This implies for any configurations σ and τ, $P(\sigma, \tau) > 0$ if and only if $P(\sigma, \tau) > 0$. Using this, the lemma can easily be verified via detailed balance. □

As referenced above, this stationary distribution can be expressed in an alternate way using weighted perimeter.

Lemma 5. *For $c = 1 + \log_\lambda \gamma$, the stationary distribution of $\mathcal{M}$ is given by*

$$\pi(\sigma) = \lambda^{-\overline{p}(\sigma,c)}/Z,$$

where $Z = \sum_{\sigma \in \Omega} \lambda^{-\overline{p}(\sigma,c)}$.

Proof. This follows immediately from the definition of $\overline{p}(\sigma, c)$. □

Theorem 1. *Consider an execution of Markov chain $\mathcal{M}$ on state space Ω , with $\lambda > 2 + \sqrt{2}$, $\gamma > 1$, and stationary distribution π, where starting configuration σ_0 has n particles. For any constant $\alpha > \frac{\log(\lambda)}{\log(\lambda) - \log(2+\sqrt{2})} > 1$, the probability that a configuration σ drawn at random from π has $\overline{p}(\sigma, 1 + \log_\lambda \gamma) > \alpha \cdot \overline{p}_{min}$ is exponentially small in n, where $\overline{p}_{min}$ is the minimum weighted perimeter of a configuration in Ω.*

Proof. This mimics the proof of α-compression in [2], though additional insights and care were necessary to accommodate the difficulties introduced by considering weighted perimeter instead of perimeter.

Given any configuration σ, let

$$w(\sigma) := \pi(\sigma) \cdot Z = \lambda^{-p(\sigma)} \gamma^{-g(\sigma)} = \lambda^{-\overline{p}(\sigma, 1+\log_\lambda \gamma)}.$$

For a set of configurations $S \subseteq \Omega$, we let $w(S) = \sum_{\sigma \in S} w(\sigma)$. Let $\sigma_{min} \in \Omega$ be a configuration of n particles with minimal weighted perimeter $\overline{p}_{min}$, and let S_α be the set of configurations with weighted perimeter at least $\alpha \cdot \overline{p}_{min}$. We show:

$$\pi(S_\alpha) = \frac{w(S_\alpha)}{Z} < \frac{w(S_\alpha)}{w(\sigma_{min})} \leq \zeta^{\sqrt{n}},$$

where $\zeta < 1$. The first equality follows from Lemma 5; the next inequality follows from the definitions of Z, w, and σ_{min}. We focus on the last inequality.

We stratify S_α into sets of configurations with the same weighted perimeter; there are at most $\mathcal{O}(n^2)$ such sets, as the total perimeter and gap perimeter can each take on at most $\mathcal{O}(n)$ values. Label these sets $A_1, A_2, \ldots, A_m$ in order of increasing weighted perimeter, where m is the total number of distinct weighted perimeters possible for configurations in S_α. Let $\overline{p}_i$ be the weighted perimeter of all configurations in set A_i; since $A_i \subseteq S_\alpha$, we have $\overline{p}_i \geq \alpha \cdot \overline{p}_{min}$.

We note $w(\sigma) = \lambda^{-\overline{p}_i}$ for every $\sigma \in A_i$, so to bound $w(A_i)$ it only remains to bound $|A_i|$. Any configuration with weighted perimeter $\overline{p}_i$ has perimeter $p \leq \overline{p}_i$, and a result from [2] which exploits a connection between particle configurations and self-avoiding walks in the hexagon lattice shows that the number of connected hole-free particle configurations with perimeter p is at most $f(p)(2+\sqrt{2})^p$, for some subexponential function f. Letting p_{min} denote the minimum possible (unweighted) perimeter of a configuration of n particles, we conclude that

$$w(A_i) \leq \lambda^{-\overline{p}_i} \cdot \sum_{p=p_{min}}^{\overline{p}_i} f(p) \left(2 + \sqrt{2}\right)^p \leq \lambda^{-\overline{p}_i} f'(\overline{p}_i) \left(2 + \sqrt{2}\right)^{\overline{p}_i},$$

where $f'(\overline{p}_i) = \sum_{p=p_{min}}^{\overline{p}_i} f(p)$ is necessarily also a subexponential function because it is a sum of at most a linear number of subexponential terms. So,

$$w(S_\alpha) = \sum_{i=1}^{m} w(A_i) \leq \sum_{i=1}^{m} f'(\overline{p}_i) \left(\frac{2+\sqrt{2}}{\lambda}\right)^{\overline{p}_i} \leq f''(n) \left(\frac{2+\sqrt{2}}{\lambda}\right)^{\alpha \cdot \overline{p}_{min}},$$

where $f''(n) = \sum_{i=1}^{m} f'(\overline{p}_i)$ is a subexponential function because $\overline{p}_i = \mathcal{O}(n)$, $m = \mathcal{O}(n^2)$, and f' is subexponential. The last inequality follows because $\lambda > 2+\sqrt{2}$ and $\overline{p}_i \geq \alpha \overline{p}_{min}$ by assumption. Finally, because $w(\sigma_{min}) = \lambda^{-\overline{p}_{min}}$,

$$\frac{w(S_\alpha)}{w(\sigma_{min})} \leq f''(n) \left(\frac{2+\sqrt{2}}{\lambda}\right)^{\alpha \cdot \overline{p}_{min}} \lambda^{\overline{p}_{min}} = f''(n) \zeta^{\overline{p}_{min}},$$

where $\zeta = \lambda \left(\frac{2+\sqrt{2}}{\lambda}\right)^{\alpha} < 1$ whenever $\alpha > \frac{\log(\lambda)}{\log(\lambda) - \log(2+\sqrt{2})}$. We have $\overline{p}_{min} \geq \sqrt{n}$ because any n particles must have perimeter at least $\sqrt{n}$. This suffices to show there is a constant $\zeta < 1$ and a subexponential function $f''(n)$ such that

$$\pi(S_\alpha) < f''(n) \zeta^{\sqrt{n}},$$

which proves the theorem. □

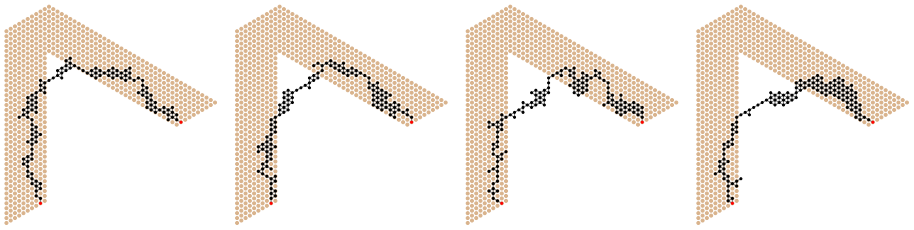
As we see in the following corollary, to solve an instance $(L, O, \sigma_0, c, \alpha)$ of the shortcut-bridging problem, one just needs to run algorithm $\mathcal{M}$ with carefully chosen parameters λ and γ.

Corollary 1. *The distributed algorithm associated with Markov chain $\mathcal{M}$ can solve any instance $(L, O, \sigma_0, c, \alpha)$ of the shortcut-bridging problem.*

Proof. It suffices to run the distributed algorithm associated with $\mathcal{M}$ starting from configuration σ_0 with parameters $\lambda > (2+\sqrt{2})^{\frac{\alpha}{\alpha-1}}$ and $\gamma = \lambda^{c-1}$. Then it holds that $\alpha > \frac{\log(\lambda)}{\log(\lambda) - \log(2+\sqrt{2})} > 1$, so by Theorem 1 the system reaches and remains with all but exponentially small probability in a set of configurations with weighted perimeter $\overline{p}(\sigma, c) \leq \alpha \cdot \overline{p}_{min}$, where $\overline{p}_{min}$ is the minimum weighted perimeter of a configuration in Ω. □

4 Dependence of Bridge Structure on Gap Angle

Specifically, we consider V-shaped land masses (e.g., Figure 2a) of various angles. We prove that our shortcut bridging algorithm exhibits a dependence on the internal angle θ of the gap that is similar to that of the army ant bridging process observed by Reid et al. [1]. When the internal angle θ is sufficiently small, with high probability the bridge constructed by the particles stays close to the bottom of the gap (away from the apex of angle θ). Furthermore, when θ is large and λ and γ satisfy certain conditions (made explicit in Theorem 3), with high probability the bridge stays close to the top of the gap. Both of these results are proven using a Peierls argument and careful analysis of the geometry of the gap. Due to space constraints, we merely state our main results and omit the proofs, while noting that they are far from trivial.



(a) (b) (c) (d)

Fig. 4: A particle system using biases $\lambda = 4$ and $\gamma = 2$ to shortcut a V-shaped land mass with $\theta = 60^\circ$ after (a) 2 million, (b) 4 million, (c) 6 million, and (d) 8 million iterations of Markov chain $\mathcal{M}$, beginning in configuration σ_0 shown in Figure 2a.

Theorem 2. *Let $\lambda > 2 + \sqrt{2}$ and $\gamma > 1$. Then there exists θ_1 such that for all $\theta < \theta_1$, the probability at stationarity of $\mathcal{M}$ that the bridge structure is strictly above the midpoint of the gap is exponentially small in n, the number of particles. In particular, $\theta_1 = 2\tan^{-1}\left(\log_{\lambda\gamma}\left(\lambda/\left(2+\sqrt{2}\right)\right)/\sqrt{3}\right)$.*

Theorem 3. *For each $\lambda > 2 + \sqrt{2}$ and $\gamma > (2+\sqrt{2})^4\lambda^4$, there is a constant $\theta_2 > 60^\circ$ such that for all $\theta \in (60^\circ, \theta_2)$, the probability at stationarity of $\mathcal{M}$ that the bridge structure goes through or below the midpoint of the gap is exponentially small in n. In particular, $\theta_2 = 2\tan^{-1}\left[\frac{1}{2\sqrt{3}}\frac{\log(\gamma\lambda^{-4})}{\log(2+\sqrt{2})} - \frac{1}{\sqrt{3}}\right]$.*

5 Simulations

In this section, we show simulation results of our algorithm running on a variety of instances. Figure 4 shows snapshots over time for a bridge shortcutting a V-shaped gap with internal angle $\theta = 60^\circ$ and biases $\lambda = 4, \gamma = 2$. Qualitatively, this bridge matches the shape and position of the army ant bridges in [1]. Figure 5 shows the resulting bridge structure when the land mass is N-shaped. Lastly, Figure 6 shows the results of an experiment which held λ, γ, and the number of iterations of $\mathcal{M}$ constant, varying only the internal angle of the V-shaped land mass. The particle system exhibited behavior consistent with the theoretical results in Section 4 and the army ant bridges, shortcutting closer to the bottom of the gap when θ is small and staying almost entirely on land when θ is large.

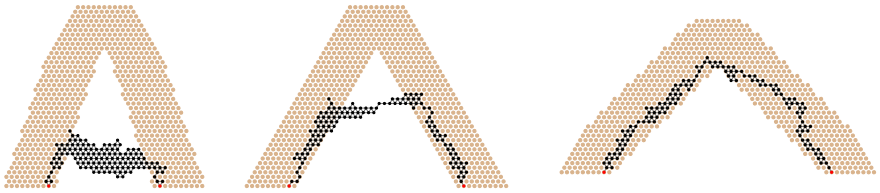
These simulations demonstrate the successful application of our stochastic approach to shortcut bridging. Moreover, experimenting with variants suggests this approach may be useful for other related applications in the future.

References

1. C.R. Reid, M.J. Lutz, S. Powell, A.B. Kao, I.D. Couzin, and S. Garnier. Army ants dynamically adjust living bridges in response to a cost–benefit trade-off. *Proceedings of the National Academy of Sciences*, 112(49):15113–15118, 2015.

(a) (b)

Fig. 5: A particle system using $\lambda = 4$ and $\gamma = 2$ to shortcut an N-shaped land mass after (a) 10 million and (b) 20 million steps of $\mathcal{M}$, beginning in σ_0 of Figure 2b.



(a) (b) (c)

Fig. 6: A particle system using biases $\lambda = 4$ and $\gamma = 2$ to shortcut a land mass with angle (a) 30°, (b) 60°, and (c) 90° after 20 million iterations of $\mathcal{M}$. For a given angle, the land mass and initial configuration were constructed as described in Section 4.

2. S. Cannon, J.J. Daymude, D. Randall, and A.W. Richa. A markov chain algorithm for compression in self-organizing particle systems. In *Proc. 2016 ACM Symposium on Principles of Distributed Computing (PODC '16)*, pages 279–288, 2016.
3. S. Camazine, K.P. Visscher, J. Finley, and S.R. Vetter. House-hunting by honey bee swarms: Collective decisions and individual behaviors. *Insectes Sociaux*, 46(4):348–360, 1999.
4. R. Jeanson, C. Rivault, J.L. Deneubourg, S. Blanco, R. Fournier, C. Jost, and G. Theraulaz. Self-organized aggregation in cockroaches. *Animal Behaviour*, 69(1):169–180, 2005.
5. A.M. Mohammed, P. Šulc, J. Zenk, and R. Schulman. Self-assembling DNA nanotubes to connect molecular landmarks. *Nature Nanotechnology*, 12:312–316, 2017.
6. D. Woods. Intrinsic universality and the computational power of self-assembly. In *Proc. Machines, Computations and Universality 2013 (MCU '13)*, pages 16–22, 2013.
7. D. Angluin, J. Aspnes, Z. Diamadi, M. J. Fischer, and R. Peralta. Computation in networks of passively mobile finite-state sensors. *Distributed Computing*, 18(4):235–253, 2006.
8. C.R. Reid and T. Latty. Collective behaviour and swarm intelligence in slime moulds. *FEMS Microbiology Reviews*, 40(6):798–806, 2016.
9. S.M. Douglas, H. Dietz, T. Liedl, B. Högberg, F. Graf, and W.M. Shih. Self-assembly of DNA into nanoscale three-dimensional shapes. *Nature*, 459:414–418, 2009.
10. K.C. Cheung, E.D. Demaine, J.R. Bachrach, and S. Griffith. Programmable assembly with universally foldable strings (moteins). *IEEE Transactions on Robotics*, 27(4):718–729, 2011.

11. B. Wei, M. Dai, and P. Yin. Complex shapes self-assembled from single-stranded DNA tiles. *Nature*, 485:623–626, 2012.
12. M. Cieliebak, P. Flocchini, G. Prencipe, and N. Santoro. Distributed computing by mobile robots: Gathering. *SIAM Journal on Computing*, 41(4):829–879, 2012.
13. P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer. Arbitrary pattern formation by asynchronous, anonymous, oblivious robots. *Theoretical Computer Science*, 407(1):412–447, 2008.
14. M. Rubenstein, A. Cornejo, and R. Nagpal. Programmable self-assembly in a thousand-robot swarm. *Science*, 345(6198):795–799, 2014.
15. B. Chazelle. Natural algorithms. In *Proc. 2009 ACM-SIAM Symposium on Discrete Algorithms (SODA09)*, pages 422–431, 2009.
16. M. Yim, W.-M. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins, and G. S. Chirikjian. Modular self-reconfigurable robot systems. *IEEE Robotics Automation Magazine*, 14(1):43–52, 2007.
17. G. Chirikjian. Kinematics of a metamorphic robotic system. In *Proc. 1994 International Conference on Robotics and Automation (ICRA '94)*, volume 1, pages 449–455, 1994.
18. J.E. Walter, J.L. Welch, and N.M. Amato. Distributed reconfiguration of metamorphic robot chains. In *Proc. 2000 ACM Symposium on Principles of Distributed Computing (PODC '00)*, pages 171–180, 2000.
19. J. E. Walter, M.E. Brooks, D.F. Little, and N.M. Amato. Enveloping multi-pocket obstacles with hexagonal metamorphic robots. In *Proc. 2004 IEEE International Conference on Robotics and Automation (ICRA '04)*, pages 2204–2209, 2004.
20. D. Woods, H.-L. Chen, S. Goodfriend, N. Dabby, E. Winfree, and P. Yin. Active self-assembly of algorithmic shapes and patterns in polylogarithmic time. In *Proc. 4th Innovations in Theoretical Computer Science Conference (ITCS '13)*, pages 353–354, 2013.
21. M. Chen, D. Xin, and D. Woods. Parallel computation using active self-assembly. *Natural Computing*, 14(2):225–250, 2015.
22. Z. Derakhshandeh, R. Gmyr, T. Strothmann, R.A. Bazzi, A.W. Richa, and C. Scheideler. Leader election and shape formation with self-organizing programmable matter. In *Proc. 21st International Conference on DNA Computing and Molecular Programming (DNA21)*, pages 117–132, 2015.
23. N. Lynch. *Distributed Algorithms*. Morgan Kauffman, 1996.
24. R.J. Baxter, I. G. Enting, and S.K. Tsang. Hard-square lattice gas. *Journal of Statistical Physics*, 22:465–489, 1980.
25. R. Restrepo, J. Shin, P. Tetali, E. Vigoda, and L. Yang. Improving mixing conditions on the grid for counting and sampling independent sets. *Probability Theory and Related Fields*, 156:75–99, 2013.
26. A. Blanca, Y. Chen, D. Galvin, D. Randall, and P. Tetali. Phase coexistence for the hard-core model on $\mathbb{Z}^2$. *Submitted*, 2017.
27. D.A. Levin, Y. Peres, and E.L. Wilmer. *Markov Chains and Mixing Times*. American Mathematical Society, 2009.
28. W. Feller. *An Introduction to Probability Theory and Its Applications, Vol. 1*. Wiley, 1968.
29. W.K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.
30. Z. Derakhshandeh, R. Gmyr, A.W. Richa, C. Scheideler, and T. Strothmann. Universal coating for programmable matter. *Theoretical Computer Science*, 671:56–68, 2017.