Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Path finding in the tile assembly model

Yuriy Brun [*], Dustin Reishus

Department of Computer Science, University of Southern California, Los Angeles, CA 90089, United States

A R T I C L E I N F O

Keywords:
Self-assembly
Tile assembly model
Amorphous computing
Swarm computing
Robotics
Crystal growth
Molecular computation
Natural computation
Path-finding

A B S T R A C T

Swarm robotics, active self-assembly, and amorphous computing are fields that focus on designing systems of large numbers of small, simple components that can cooperate to complete complex tasks. Many of these systems are inspired by biological systems, and all attempt to use the simplest components and environments possible, while still being capable of achieving their goals. The canonical problems for such biologically-inspired systems are shape assembly and path finding. In this paper, we demonstrate path finding in the well-studied tile assembly model, a model of molecular self-assembly that is strictly simpler than other biologically-inspired models. As in related work, our systems function in the presence of obstacles and can be made fault-tolerant. The path-finding systems use $\Theta(1)$ distinct components and find minimal-length paths in time linear in the length of the path.

1. Introduction

Swarm robotics, active self-assembly, and amorphous computing are fields that focus on designing systems of small, simple components that are capable of cooperating to complete complex tasks. Many of these systems have been inspired by biological systems seen in nature, so we will refer to them as biologically-inspired systems. Work on biologically-inspired systems started in theoretical explorations [1,8,18,25,30] and fueled the creation of distributed robotic systems in hardware, in which individual robots with limited capabilities come together in swarms to exhibit complex emergent behaviors [15,27,37]. The objective of the majority of the theoretical work has been to design systems that perform the most complex tasks using the simplest components. Because systems are built out of simple, and therefore cheap, components, creating a large number of components is typically not a concern, but a large number of distinct components is. To further reduce the cost of the component-manufacturing process, many of the systems strive to allow for unreliable components. In large, much of the work in these fields is inspired by biological systems that not only build complex systems out of simple and cheap components, but that also often deal with faulty and malicious agents.

Biologically-inspired systems are typically made up of a large number of identical components that are resource- and computational-power-limited agents. Various researchers have defined distinct models for studying such systems; the models differ in the components, in the types of interactions between components, and in the environmental resources available to the components.

The primary goal behind the creation of many of these models is to use the simplest components to achieve complex behavior such as the assembly of shapes or formation of paths between points. While these may not seem like complex tasks on the first inspection, these behaviors can be used as primitives to accomplish more practical results. For example, the path-forming primitive can be used to form wires between two electrodes on a surface.

It is not uncommon for new fields to remain heterogeneous for some time, with every research group coming up with its own definitions of systems, tasks, and components. However, as the common goal of biologically-inspired systems is to use

[*] Corresponding author.
E-mail addresses: ybrun@usc.edu (Y. Brun), reishus@usc.edu (D. Reishus).

0304-3975/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2008.12.008

the simplest and cheapest components to produce systems that complete the most complex tasks, it is reasonable to compare the models in which these systems reside. Unfortunately, such a comparison is unlikely to produce a single clear winner. For example, some models may use low-memory components but require complex communication abilities, others may require more memory but use simpler communication, and others may require minimal communication and memory but rely on careful control of the components' motions. Further, the complexity of the tasks is an ambiguous quantity: is it harder to find a path between points or assemble a square? In this paper, we show that there are systems in an extremely simple model, whose components have practically no memory, communication, or control abilities, capable of accomplishing some of the tasks commonly presented in related literature.

To this end, we leverage the tile assembly model [42,41,36], a formal model of crystal growth. It was designed to model self-assembly of molecules such as DNA, and thus its components are no more complex than oversimplified biological molecules. In essence, a component is a square with a label on each of its four sides. Components can neither change their labels nor input or output any information. They can, however, attach to other components when the labels on their sides match. The tile assembly model is a formal mathematical model that allows for the study of assembly time and construction complexity. Many other biologically-inspired models lack the formalism to allow this type of study. We will fully define the tile assembly model in Section 3, then present tile systems that find paths between points, and show that these systems exhibit the same robustness demonstrated by other biologically-inspired systems.

2. Related work

The idea of using swarms of simple, cheap, unreliable, and tiny (invisible to the naked eye) components to solve complex problems is attractive because manufacturing such components can be automated and their execution may take place in extreme environments. In one exploration of computing devices, Abelson et al. [1] formally defined an amorphous computer to be a 2D sheet with randomly placed immobile robots. The robots have wireless communication power with radius far smaller than the sheet, and their computational abilities are restricted to be less powerful than Turing machines, but are otherwise left open. In general, these robots are expected to have some memory and a finite control. This definition formed a medium for researchers to test the power of simple components and to experiment with programming those components to complete complex tasks. The path-finding problem in this model was solved in [15] with the use of messages similar to chemical gradients used in biological systems.

Nagpal showed that systems in the amorphous computer model, leveraging the primitive path-finding operation, can create complex shapes [30]. Later, Clement et al. [18] improved the path-finding procedure to be robust to robot failure.

Meanwhile, other researchers developed their own models, similar to the amorphous computer. Arbuckle et al. [8] concentrated on keeping the robots to have only a few bits of memory, and allow the robots to move around a 2D surface. They have demonstrated the ability to build paths, assemble shapes, and repair formed paths and shapes. Others attempted to implement physical manifestations of these models [15,27,37].

While a wealth of literature exists on biologically-inspired systems, this literature lacks the organization and common definitions necessary to effectively compare the complexity of the basic components or the complexity of the tasks performed. Our systems presented in this paper use components far simpler than the ones described above (they have no finite control, minimal read-only memory, and incredibly limited communication abilities), and perform some of the same tasks we have described thus far.

To demonstrate how such simple components can perform complex tasks, we leverage the field of self-assembly. Winfree showed that DNA computation is Turing-universal [40]. While DNA computation suffers from relatively high error rates, the study of self-assembly shows how to utilize redundancy to design systems with built-in error correction [44,10,17,31,43]. Researchers have used DNA to assemble crystals with patterns of binary counters [9] and Sierpinski triangles [35].

The tile assembly model [42,41,36] is a formal model of crystal growth. It was designed to model self-assembly of molecules such as DNA. It is an extension of a model proposed by Wang [39]. The model was fully defined by Rothemund and Winfree [36], and the definitions here are similar to those, but we restate them here for completeness and to assist the reader. Intuitively, the model has *tiles* or squares that stick or do not stick together based on various binding domains on their four sides. Each tile has a binding domain on its north, east, south, and west side, which define the type of the tile. In this definition, tiles are not allowed to rotate. A tile may stick to another tile when the binding domains on the abutting sides of those tiles match and the total strength of all the binding domains on that tile exceeds the current temperature.

Winfree showed that the tile assembly model at temperature 2 is Turing-universal [41] by showing that a tile system can simulate Wang tiles [39], which are universal [11,34]. We have previously shown that the problem of whether a given set of tiles can form an infinite path is undecidable [6]. This work implies that the tile assembly model at temperature 1 is Turing-universal.

Two important questions about self-assembling systems in the tile assembly model are: "what is a minimal tile set that can accomplish a particular goal?" and "what is the minimum assembly time for that system?" Here, we study systems that find paths and answer these questions about our systems. Adleman has emphasized studying the number of steps it takes for an assembly to complete (assuming maximum parallelism) and the minimal number of tiles necessary to assemble a shape [2]. He answered these questions for n-long linear polymers [4]. Previously, we have extended these questions to apply to systems that compute functions, rather than assemble shapes, deterministically [12] and nondeterministically [13,14].

One of the potential applications of the tile assembly model is self-assembling electronic circuits [19,32]. Researchers have shown that it is possible to attach simple, electrically-active components to DNA tiles and use the self-assembling interactions of the tiles to arrange these components [45]. One of the most basic tasks one might want to use self-assembly to complete is constructing a wire between two points. This task involves finding a path between those points. One might further specify that the path should be short, use no extraneous components, or perhaps avoid certain regions (other circuit elements, for example). We will present systems that solve these tasks.

The domino snake problem is a variant of the path-finding problem. It asks, given an arbitrary set of tiles and two points in the plane, is it possible to construct a path between the points using those tiles? This problem turns out to be decidable [29, 22]. However, if the path is restricted to a half-plane or a quadrant, it becomes undecidable [20]. In fact, the problem remains undecidable even if there is only one point in the plane through which the path is not allowed to travel [21].

We proposed and studied systems that compute the sums and products of two numbers using the tile assembly model [12]. We found that in the tile assembly model, adding and multiplying can be done using $\Theta(1)$ tiles (as few as 8 tiles for addition and as few as 28 tiles for multiplication), and that both computations can be carried out in time linear in the input size. We then showed that systems can be combined to create systems with more complex behavior, and designed systems that factor numbers [13] and solve NP-complete problems [14]. Other early attempts at nondeterministic computation include a proposal by Lagoudakis et al. [26] to solve the satisfiability problem. They informally define a system that uses $\Theta(n^2)$ distinct tiles to nondeterministically compute whether or not an n-variable boolean formula is satisfiable.

Barish et al. [9] have demonstrated DNA implementations of tile systems, one that copies an input and another that counts in binary. Similarly, Rothemund et al. [35] have demonstrated a DNA implementation of a tile system that computes the *xor* function, resulting in a Sierpinski triangle. These systems grow crystals using double-crossover complexes [23] as tiles.

Adleman proposed studying the complexity of tile systems that can uniquely produce $n \times n$ squares. A series of researchers [36,3,5,28] proceeded to answer the questions: “what is a minimal tile set that can assemble such shapes?” and “what is the assembly time for these systems?” They showed that the minimal tile set that assembles $n \times n$ squares is of size $O(\frac{\log n}{\log\log n})$ and the optimal assembly time is $\Theta(n)$ [5].

Researchers have also studied variations of the traditional tile assembly model. Aggarwal et al. and Kao et al. have shown that changing the temperature of assembly from a constant throughout the assembly process to a discrete function reduces the minimal tile set that can build an $n \times n$ square to a size $\Theta(1)$ tile set [7,24]. In our work with path-finding systems we allow the temperature to change once.

3. Tile assembly model

Formally, let Σ be a finite alphabet of symbols called **binding domains** such that $null \in \Sigma$. We will always assume $null \in \Sigma$ even when we do not specify so explicitly. A **tile** over a set Σ of binding domains is a 4-tuple $\langle \sigma_N, \sigma_E, \sigma_S, \sigma_W \rangle \in \Sigma^4$. A **position** is an element of $\mathbb{Z}^2$. The set of directions $D = \{N, E, S, W\}$ is a set of four functions from positions to positions, i.e. $\mathbb{Z}^2$ to $\mathbb{Z}^2$, such that for all positions (x, y), $N(x, y) = (x, y + 1)$, $E(x, y) = (x + 1, y)$, $S(x, y) = (x, y - 1)$, and $W(x, y) = (x - 1, y)$. The positions (x, y) and (x', y') are neighbors iff $\exists d \in D$ such that $d(x, y) = (x', y')$. For a tile t, for $d \in D$, we will refer to $bd_d(t)$ as the binding domain of tile t on d's side. A special tile $empty = \langle null, null, null, null \rangle$ represents the absence of all other tiles.

A **strength function** $g : \Sigma \times \Sigma \to \mathbb{N}$, where g is commutative and $\forall \sigma \in \Sigma\ g(null, \sigma) = 0$, denotes the strength of the binding domains.

Let T be a set of tiles containing *empty*. A **configuration** of T is a function $A : \mathbb{Z} \times \mathbb{Z} \to T$. We write $(x, y) \in A$ iff $A(x, y) \neq empty$. A is finite iff there is only a finite number of distinct positions $(x, y) \in A$. When it does not lead to ambiguity, we will sometimes say tile t is in position (x, y) if $A(x, y) = t$, and we will refer to the position a tile is in implicitly (e.g., tile t_1 neighbors tile t_2).

Finally, a **tile system** $\mathbb{S}$ is a triple $\langle T, g, \tau \rangle$, where $T \subseteq \Sigma^4$ is a finite set of tiles over a set Σ of binding domains such that $empty \in T$, $g : \Sigma \times \Sigma \to \mathbb{N}$ is a strength function, and $\tau \in \mathbb{N}$ is the temperature.

If $\mathbb{S} = \langle T, g, \tau \rangle$ is a tile system and A is a configuration of some set of tiles $T' \subseteq \Sigma^4$ then a tile $t \in T$ can **attach** to A at position (x, y) and produce a new configuration A' iff:

- $(x, y) \notin A$, and
- $\sum_{d \in D} g(bd_d(t), bd_{d^{-1}}(A(d(x, y)))) \geq \tau$, and
- $\forall (u, v) \in \mathbb{Z}^2, (u, v) \neq (x, y) \Rightarrow A'(u, v) = A(u, v)$, and
- $A'(x, y) = t$.

That is, a tile can attach to a configuration only in empty positions and only if the total strength of the appropriate binding domains on the tiles in neighboring positions meets or exceeds the temperature τ. For example, if for all σ, $g(\sigma, \sigma) = 1$ and $\tau = 2$ then a tile t can attach only at positions with matching binding domains on the tiles in at least two neighboring positions.

Alternatively, if $\mathbb{S} = \langle T, g, \tau \rangle$ is a tile system and A is a configuration of some set of tiles $T' \subseteq \Sigma^4$ then a tile $t \in T$ can **detach** from A at position (x, y) and produce a new configuration A' iff:

- $(x, y) \in A$, and
- $\sum_{d \in D} g(bd_d(t), bd_{d^{-1}}(A(d(x, y)))) < \tau$, and

- $\forall (u, v) \in \mathbb{Z}^2, (u, v) \neq (x, y) \Rightarrow A'(u, v) = A(u, v)$, and
- $(x, y) \notin A'$.

That is, a tile can detach from a configuration only if the total strength of the appropriate binding domains on the tiles in neighboring positions is less than the temperature τ. For example, if for all $\sigma, g(\sigma, \sigma) = 1$ and $\tau = 2$ then a tile t can detach from every position with matching binding domains on the tiles in fewer than two adjacent positions.

Given a tile system $\mathbb{S} = \langle T, g, \tau \rangle$, a set of tiles Γ, and a **seed configuration** $S_0 : \mathbb{Z}^2 \to \Gamma$, if the above conditions are satisfied, one may attach and detach tiles of T to and from S_0. Note that we allow the codomain of S_0 to be Γ, a set of tiles which may be different from T. Let $V_0 \subseteq \mathbb{Z}^2$ be the set of all positions where a tile from T can detach from S_0. Let S_0' be the configuration produced by all the tiles detaching from S_0 at all positions in V_0. Let $W_0 \subseteq \mathbb{Z}^2$ be the set of all positions where at least one tile from T can attach to S_0'. For all $w \in W_0$ let U_w be the set of all tiles that can attach to S_0' at w. Let $\hat{S}_1$ be the set of all configurations S_1 such that for all positions $p \in S_0', S_1(p) = S_0'(p)$ and for all positions $w \in W_0, S_1(w) \in U_w$ and for all positions $p \notin S_0 \cup W_0, S_1(p) = empty$. For all $S_1 \in \hat{S}_1$ we say that $\mathbb{S}$ **produces** S_1 on S_0 in one step. If $A_0, A_1, \ldots, A_n$ are configurations such that for all $i \in \{1, 2, \ldots, n\}$, $\mathbb{S}$ produces A_i on A_{i-1} in one step, then we say that $\mathbb{S}$ produces A_n on A_0 in n steps. When the number of steps taken to produce a configuration is not important, we will simply say $\mathbb{S}$ produces a configuration A on a configuration A' if there exists $k \in \mathbb{N}$ such that $\mathbb{S}$ produces A on A' in k steps. If the only configuration produced by $\mathbb{S}$ on A is A itself, then A is said to be a **final configuration**. If there is only one final configuration A produced by $\mathbb{S}$ on S_0, then $\mathbb{S}$ is said to produce a **unique** final configuration on S_0. Finally, if A is a final configuration produced by $\mathbb{S}$ on S_0 and n is the least integer such that A is produced by $\mathbb{S}$ on S_0 in n steps, then n is the **assembly time** of $\mathbb{S}$ on S_0 to produce A.

Less formally, if $\mathbb{S}$ is a tile system and S_0 is a seed configuration, one may detach and attach tiles of T from and to S_0. If one detaches all the tiles one can from S_0 and then attaches all the tiles one can, one creates a configuration S_1 that is produced by $\mathbb{S}$ on S_0 in one step. Note that there may have been many choices of which tile to attach in each position. This process can be repeated until no detachments or attachments are possible, at which point one is left with a final configuration. The number of steps taken to produce the final configuration is its assembly time.

Sometimes, it may be useful to allow a system to work at a certain temperature and then later increase that temperature. This process would allow a system to "explore" the space of configurations and then melt away unsuccessful explorations. Thus we define a **melting** tile system $\mathbb{S} = \langle T, g, \tau_g, \tau_m \rangle$. Let R be the set of configurations that the tile system $\langle T, g, \tau_g \rangle$ produces on a seed configuration S_0 in t steps. Let F be the union of the sets of final configurations that the tile system $\langle T, g, \tau_m \rangle$ produces on each configuration of R, where tiles are forbidden from detaching from positions in S_0. We say that $\mathbb{S}$ produces the final configurations F on S_0 with the **switch time** t. If F is a singleton, then we say that $\mathbb{S}$ produces a unique final configuration F on S_0 with the switch time t.

4. Path finding

In this section, we will first discuss a tile system that finds paths in an unconstrained environment in Section 4.1, and then a tile system that can handle obstacles in Section 4.2.

Path finding is the problem of forming a path between two points on a 2D plane. Intuitively, given a seed configuration with a single start and a single goal tile, a path-finding system should attach tiles to connect the start to the goal. It is straightforward to design such a system that leaves extraneous tiles because one could just fill the plane with fully-connected tiles and claim that the path is there. Thus we wish to restrict systems to leave no extraneous tiles in the final configuration.

Formally, an **untiled path** of length n is a sequence of positions $\langle p_1, p_2, \ldots, p_n \rangle$ such that $\forall i \neq j \in \{1, 2, \ldots, n\}, p_i \neq p_j$ and $\forall i \in \{1, 2, \ldots, n-1\}, \exists d \in D$ such that $p_{i+1} = d(p_i)$. Given a tile system $\mathbb{S} = \langle T, g, \tau \rangle$ and a configuration A of some set of tiles T', a **tiled path**, or simply a **path**, in A of length n is an untiled path P of length n such that for all $i \in \{2, 3, \ldots, n-1\}$, $A(p_i)$ is attached to $A(p_{i-1})$ with strength $a > 0$ and $A(p_i)$ is attached to $A(p_{i+1})$ with strength $b > 0$ and $a + b \geq \tau$.

That is to say, an untiled path is a sequence of neighboring positions that does not cross itself. A path in a configuration A is a sequence of neighboring positions such that the tile in each of the positions (except the first and the last) is attached to the preceding and succeeding tiles with total strength at least the temperature. In some related literature [6], what we call paths have been called ribbons.

We say that a melting tile system $\mathbb{S}$ solves the path-finding problem if for all seed configurations $S_0 : \mathbb{N}^2 \to \Gamma$, where $\{\mathsf{S}, \mathsf{G}\} \subseteq \Gamma$ and S_0 maps exactly one position to S and exactly one position to G, there exists $t \in \mathbb{N}$ such that for all final configurations F that $\mathbb{S}$ produces on S_0 with the switch time t there is a path $\langle p_0, p_1, p_2, \ldots, p_n \rangle$ such that:

- $F(p_0) = \mathsf{S}$,
- $F(p_n) = \mathsf{G}$,
- $\forall (x, y) \in F$, (x, y) is on a path in F of minimal length.

Informally, starting from a configuration with a single S and G, $\mathbb{S}$ must produce a final configuration F that contains a path of connected tiles from S to G of minimal length, and every position in F must be on such a path. For now, our definition of "minimal length" from p_0 to p_n will be the L^1 norm or Manhattan distance between the two positions.

The path-finding problem is analogous to the path-finding relatives that researchers have solved previously [1,8,18,30]. Demonstrating that there exists a tile system that solves the path-finding problem indicates that it can be solved with simpler basic components than those used in the related work. We now present such a system.

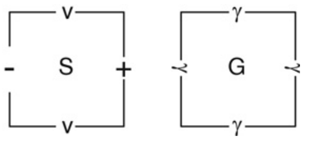

Fig. 1. The start (S) and goal (G) tiles of $\mathbb{S}_{dpf}$.

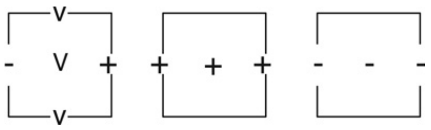

Fig. 2. The three tiles of T_{dpf}.

4.1. Direct path finding

The intuition behind the first system we present is the following: tiles will grow vertically from the start tile, in both directions (north and south). Each of the tiles that attaches vertically will have tiles attach horizontally, searching for the goal tile. Once a path is found (we discuss below how one might know when such a path is found), we increase the temperature and all non-successful paths melt away. We call this system the direct path-finding system, or $\mathbb{S}_{dpf}$.

$\mathbb{S}_{dpf}$ will contain tiles over the following set of binding domains: $\Sigma_{dpf} = \{null, \gamma, \mathsf{v}, +, -\}$. Fig. 1 shows Γ_{dpf}, which contains the start (S) and goal (G) tiles. The start tile has v binding domains on its north and south and a $+$ binding domain on its east and $-$ binding domain on its west, while the goal tile is covered with γ binding domains.

Fig. 2 shows the three tiles of T_{dpf}. The V tile is designed to attach vertically to the S tile and the other two tiles ($+$ and $-$) are designed to attach horizontally to S and the V tiles. These two tiles have *null* glues on their north and south sides (unlabeled in the figure). The glue strength function g_{dpf} is defined as follows:

- $\forall \sigma \in \Sigma_{dpf}, g_{dpf}(\sigma, null) = g_{dpf}(null, \sigma) = 0$,
- $\forall \sigma \in \Sigma_{dpf} \setminus \{null\}, g_{dpf}(\sigma, \gamma) = g_{dpf}(\gamma, \sigma) = 1$,
- $\forall \sigma \in \Sigma_{dpf} \setminus \{\gamma, null\}, g_{dpf}(\sigma, \sigma) = 2$,
- $\forall \sigma, \sigma' \in \Sigma_{dpf} \setminus \{\gamma\}$, if $\sigma \neq \sigma'$ then $g_{dpf}(\sigma, \sigma') = 0$.

Intuitively, the *null* binding domain has binding strength 0 to every other binding domain; the γ binding domain has binding strength 1 to every other binding domain, except *null*; and all other binding domains have binding strength 2 to themselves and 0 to others.

We will show that the melting tile system $\mathbb{S}_{dpf} = \langle T_{dpf}, g_{dpf}, 2, 3\rangle$ solves the path-finding problem. Let us examine the possible attachments of tiles to a seed configuration S_0 that maps one position to S and one other position to G. Fig. 3(a) shows one sample seed configuration. Because the temperature during the first stage of the process is 2 and all of G's binding domains are γ and $\forall \sigma \in \Sigma_{dpf}, g_{dpf}(\sigma, \gamma) \leq 1$, unless some tile is G's neighbor's neighbor (there is a gap of one tile between them, thus allowing a tile to attach to them both), no tile may attach to G. Tiles may, however, attach to S. In the first time step, four tiles will attach to S, one on each of its sides. Fig. 3(b) shows the configuration after one time step. Tiles will continue to attach and grow larger diamond formations around the S tile. At step t, all of the positions with Manhattan distance t from S will be filled with tiles attached to the growing configuration. Fig. 3(c) shows the configuration after two steps. Eventually, the diamond will grow large enough to touch the G tile. Fig. 3(d) shows part of a configuration that has reached G. Note that only a single tile actually attaches to G, the tile directly to its west; the adjacent tile to its south has a *null* north binding domain and thus the strength of the bond between it and G is 0.

At this point, the tile system temperature can be increased to 3. Note that the tile directly west of G is attached with strength 3, thus it will remain in the structure. The tiles highlighted in gray in Fig. 3(e) are only attached to the assembly with strength 2, and thus will detach. Once detached, some tiles that were previously connected by two strength 2 binding domains will now be only connected by a single such domain, and thus will detach (Fig. 3(f)). This process will continue until the only tiles left are the tiles in the path going north or south from S until G's row and then going east or west to G.

Theorem 1. *The melting tile system $\mathbb{S}_{dpf} = \langle T_{dpf}, g_{dpf}, 2, 3\rangle$ solves the path-finding problem.*

Proof. Let S_0 be a seed configuration such that $S_0(x_S, y_S) = \mathsf{S}$ and $S_0(x_G, y_G) = \mathsf{G}$ and for all other positions (x, y), $S_0(x, y) = empty$. Let $d = |x_S - x_G| + |y_S - y_G|$ be the Manhattan distance from S to G. We will show by induction that after $t < d$ steps:

(1) All positions with Manhattan distance $\leq t$ from (x_S, y_S) will not be *empty*.
(2) All positions with Manhattan distance $> t$ from (x_S, y_S) will be *empty* (except for (x_G, y_G)).
(3) The north binding domain of the tile in position $(x_S, y_S + t)$ will be v.
(4) The south binding domain of the tile in position $(x_S, y_S - t)$ will be v.
(5) All other exposed north and south binding domains will be *null* (except for those on G).
(6) All exposed east binding domains will be $+$ (except for those on G).

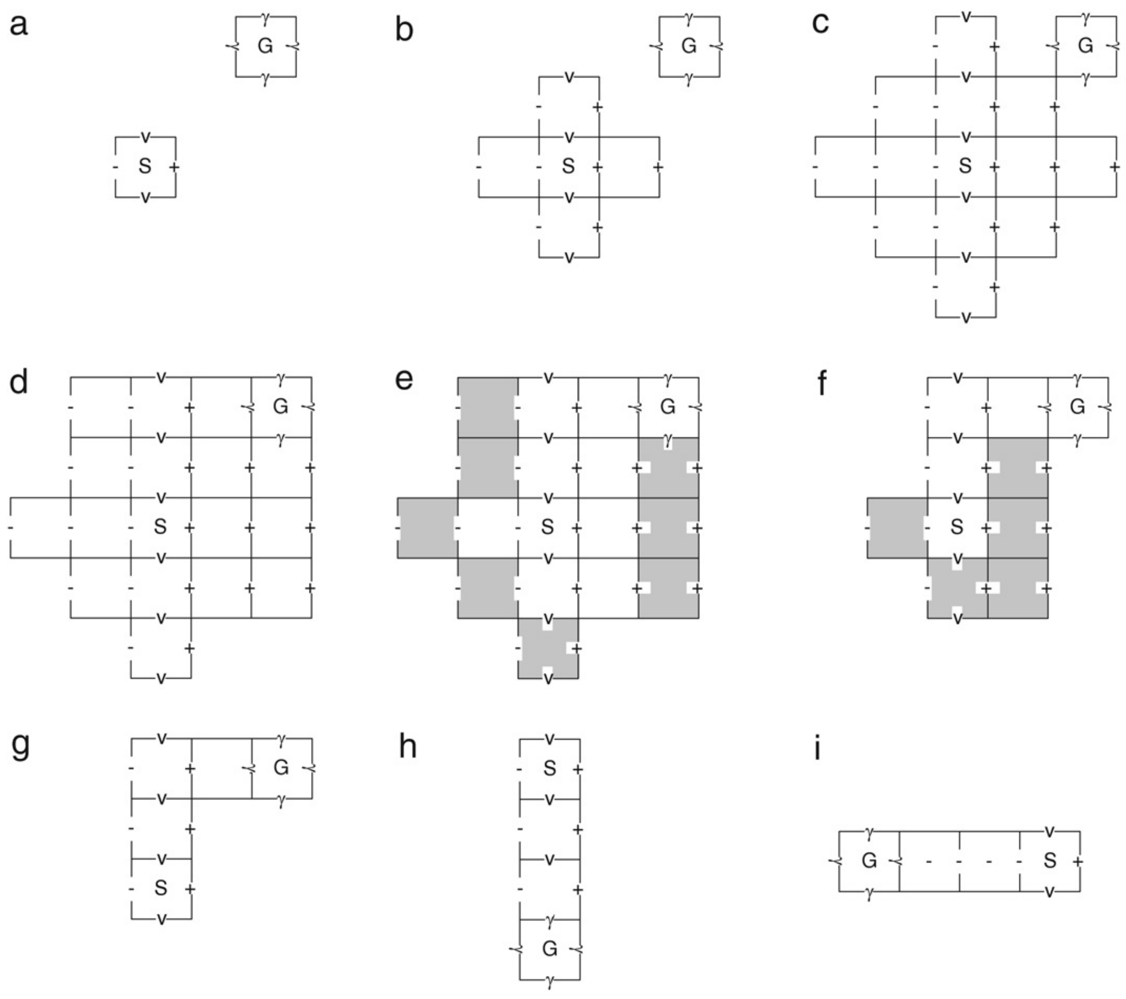

Fig. 3. An example execution of $\mathbb{S}_{dpf}$ (a–g). $\mathbb{S}_{dpf}$ can find paths between S and G in different columns and rows (g), in the same column (h), and in the same row (i).

(7) All exposed west binding domains will be $-$ (except for those on G).

(8) If $t > 0$, then all tiles with Manhattan distance t from (x_S, y_S) will be attached to exactly one tile with Manhattan distance $t-1$ from (x_S, y_S).

At $t = 0$, the configuration is S_0 and the only position with Manhattan distance 0 from S is S itself, thus all the conditions are satisfied.

Let S_t be a configuration produced by $\mathbb{S}_{dpf}$ on S_0 in $t < d$ steps. Assume S_t satisfies the above conditions. Note that G has γ binding domains on every side, and $g_{dpf}(\gamma, \sigma) = 1$ for all $\sigma \in \Sigma_{dpf} \setminus \{null\}$, so at temperature 2 no tile can attach to G without also attaching to at least one other tile. Further, since all of the binding domains on tiles from T attach to S and to other tiles from T with strength 2, if a tile from T can attach to G at temperature 2, then it would also be able to attach in that position if G were not there.

Since tiles can only attach to S_t in positions neighboring a position in S_t, and by (2) all positions with Manhattan distance $>t$ from (x_S, y_S) will be *empty* (except for (x_G, y_G)), in step $t+1$ no tile will be able to attach to S_t in a position with Manhattan distance $>t+1$ from (x_S, y_S), so S_{t+1} will satisfy condition (2). By (1) and the fact that every position with Manhattan distance $t+1$ from (x_S, y_S) neighbors a position with Manhattan distance t, and by (3), (4), (6), and (7), each of those positions has at least one neighboring tile with an exposed strength 2 binding domain, we know that in step $t+1$ at least one tile will be able to attach in every position with Manhattan distance $t+1$ from (x_S, y_S), so S_{t+1} will satisfy condition (1).

By (2), we know that position $(x_S, y_S + t + 1)$ (respectively position $(x_S, y_S - t - 1)$) is not in S_t. The only neighboring position with Manhattan distance $\leq t$ is $(x_S, y_S + t)$ (resp. $(x_S, y_S - t)$). By (3) (resp. (4)), we know the north (resp. south) binding domain of the tile in that position is v. The only tile that can attach to S_t in position $(x_S, y_S + t + 1)$ (resp. position $(x_S, y_S - t - 1)$) is the V tile. When this tile attaches, it will expose a north (resp. south) v binding domain, an east $+$ binding domain, and a west $-$ binding domain, so S_{t+1} will satisfy conditions (3) and (4).

Every position (x, y) with Manhattan distance $t+1$ from (x_S, y_S) except for $(x_S, y_S + t + 1)$ and $(x_S, y_S - t - 1)$ neighbors a position (x', y') with Manhattan distance t from (x_S, y_S) on either the east or the west side. By (2), $S_t(x, y) = empty$. By (6) (respectively (7)), all the exposed binding domains on the east will be $+$ (resp. west will be $-$). The only tile that can attach in position (x, y) is $+$ (resp. $-$). That tile will expose a $+$ (resp. $-$) binding domain on the east (resp. west) and *null* binding domains on the north and south. Therefore S_{t+1} will satisfy conditions (5), (6), and (7). If any positions in S_t neighbor (x, y) on the north or south, the abutting binding domain will be *null* by (5), so S_{t+1} satisfies condition (8).

Consider the configuration S_{d-1} produced by $\mathbb{S}_{dpf}$ on S_0 in $d - 1$ steps. The Manhattan distance from S to G is d, and all positions with Manhattan distance $d - 1$ from S will not be *empty*, so at least one position that neighbors G will be in S_{d-1}. If $x_S = x_G$ then by conditions (3) and (4), the tile that neighbors G has a v binding domain on the side that abuts G. Since $g_{dpf}(\mathrm{v}, \gamma) = 1$, the tile is attached to G with a strength 1 binding domain.

If $x_S \neq x_G$ then either $E(x_G, y_G) \in S_{d-1}$ or $W(x_G, y_G) \in S_{d-1}$. By conditions (6) and (7) a tile that neighbors G has either a $+$ or $-$ binding domain on the side that abuts G. Since $g_{dpf}(+, \gamma) = g_{dpf}(-, \gamma) = 1$, the tile is attached to G with a strength 1 binding domain. Note that only one tile in S_{d-1} can be attached to G because, by condition (5), the north and south binding domains of a tile that neighbors G on the south or north must be *null*.

Choose the switch time to be $d - 1$. Consider the tiles in positions (x, y) such that $(x, y) \in S_{d-1}$ and $(x, y) \notin S_{d-2}$. Except for the one tile t_{d-1} that is attached to G, all these tiles are attached to exactly one other tile with a strength 2 binding domain by condition (8). At temperature 3, all these tiles will detach. The one tile that is attached to G with a strength 1 binding domain is also attached to one other tile with a strength 2 binding domain (by (8)) and is thus attached to S_{d-1} with strength 3 so it will not detach at temperature 3. Since $G \in S_0$ it is forbidden from detaching.

In the next step, all the tiles in positions (x, y) such that $(x, y) \in S_{d-2}$ and $(x, y) \notin S_{d-3}$ will detach, except for the tile t_{d-2} that is attached to t_{d-1} because it is also attached to a tile in S_{d-3}. It follows that for all $k \in \{1, 2, \ldots, d-2\}$, every tile in S_k that is not in S_{k-1} will detach except for exactly one tile t_k that is attached to one tile in S_{k-1} and one tile in S_{k+1}. After $d - 1$ steps, no more tiles will be able to detach and we will be left with a final configuration F. The sequence of positions given by the position of the tiles $\langle \mathrm{S}, t_1, t_2, \ldots, t_{d-2}, t_{d-1}, \mathrm{G} \rangle$ is a path that starts at S and ends at G. Further, every position in F is on this path. Thus $\mathbb{S}_{dpf}$ solves the path-finding problem. □

Corollary 1. *Let S_0 be a seed configuration such that $S_0(x_S, y_S) = \mathrm{S}$ and $S_0(x_G, y_G) = \mathrm{G}$ and for all other positions (x, y), $S_0(x, y) = empty$. Let d be the Manhattan distance between (x_S, y_S) and (x_G, y_G). Then $\mathbb{S}_{dpf}$ produces a unique final configuration F on S_0 with switch time $d - 1$ and assembly time $\Theta(d)$. Further, for all $t \geq d$, $\mathbb{S}_{dpf}$ produces F on S_0 with switch time t.*

Proof. By conditions (3), (4), (5), (6), and (7) above, and by inspection of T_{dpf} it is clear that at each step $t < d$ at most one tile may attach in each position. After switch time $d - 1$, since the temperature is 3 no new tiles may attach, and at step $d \leq t \leq 2(d - 1)$ all tiles (except one) with Manhattan distance $2d - t - 1$ from S detach, so at after step $2(d - 1)$ no more tiles can detach. Thus $\mathbb{S}$ produces a unique final configuration with assembly time $2(d - 1) = \Theta(d)$.

Now consider a switch time $t \geq d$. At step $d - 1$, one tile attaches to G. If that tile is V, then $x_G = x_S$. If $y_G > y_S$ (respectively $y_G < y_S$) then no tile will ever be able to attach in or above (resp. in or below) row y_G. Therefore no other tile may attach to G. After step t, all tiles in positions not between S and G will eventually detach, leaving only those tiles on the path from S to G.

At step $d - 1$, if tile $+$ (respectively tile $-$) attaches to G, then $x_G > x_S$ (resp. $x_G < x_S$). If a tile attaches in position $(x_G, y_G + 1)$ or $(x_G, y_G - 1)$ it must be a $+$ tile (resp. $-$ tile). These tiles have *null* binding domains on their north and south sides, so they do not attach to G. Further, no tile will ever be able to attach in position $(x_G + 1, y_G)$ (resp. $(x_G - 1, y_G)$). Therefore, after step t, only one tile is attached to G. All the other tiles not on the path from S to G will eventually detach.

Hence, the final configuration produced by $\mathbb{S}$ on S_0 with switch time $t \geq d$ is the same as the final configuration produced by $\mathbb{S}$ on S_0 with switch time $d - 1$. □

In summary, $\mathbb{S}_{dpf}$ is guaranteed to find the shortest possible path between two points, in time linear in the length of the path.

In light of Corollary 1, one need not know the minimum switch time. However, in practice it may be helpful to be able to assess how long one has to wait before increasing the temperature. As Corollary 1 indicates, the switch time is $\Theta(d)$, where d is the Manhattan distance between S and G. More specifically, the proper minimum switch time is exactly $d - 1$. In practice, if the distance d is known, one can wait that long to increase the temperature. If d is unknown, one can devise an algorithm of increasing and decreasing the temperature repeatedly, perhaps increasing the switch time exponentially, such that in expectation a path can be found quickly.

Some biologically-inspired systems that find paths between points can recover from damage done to an existing path [8, 18]. The mechanism those systems employ are generally a detection phase (every component checks to make sure its neighbors are alive) and a repair phase (detach if your neighbor is dead, and rebuild the path from scratch). $\mathbb{S}_{dpf}$ can employ a similar mechanism: if some of the tiles are ever removed from a well-formed path, the rest of the tiles will detach automatically at temperature 3. Lowering the temperature to 2 restarts the assembly process, rebuilding the path.

4.2. Constrained path finding

Many of the biologically-inspired systems from related literature are capable not only of self-organizing to find a path between two points, but also of doing so with obstacles in the way [1,8,18,30]. In fact, the amorphous computer, and similar systems, have no sense of a 2D grid, despite having components positioned in a 2D space. The notion of a straight line

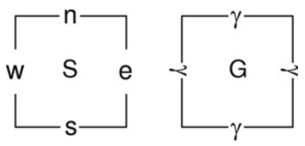

Fig. 4. The start (S) and goal (G) tiles of $\mathbb{S}_{cpf}$.

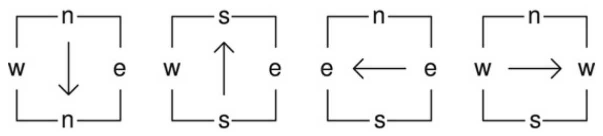

Fig. 5. The four tiles of T_{cpf}.

in such a system is simply the shortest communication path between two points, and thus if there are obstacles or regions unpopulated by robots, the system automatically "bends" the 2D space to represent straight lines as curved lines that happen to be shortest communication paths. This feature comes naturally to most such systems specifically because the systems are biologically-inspired and in biology, the individual components (e.g., cells) are not aware of their 2 or 3D grid coordinates a priori, and signals (e.g., chemical gradients) travel around obstacles.

The melting tile system $\mathbb{S}_{dpf}$ finds paths between points without obstacles. If an obstacle were to be placed in the standard path, the system would simply hit the obstacle and get stuck. In this section, we present a system, $\mathbb{S}_{cpf}$, that finds paths even with obstacles in the way.

We will denote obstacles as part of the seed configuration mapping positions with obstacles to special tile *obstacle* $= \langle \mathsf{x}, \mathsf{x}, \mathsf{x}, \mathsf{x} \rangle$. Like the special *empty* tile, if there is a configuration A and a position p such that $A(p) = obstacle$, we will say $p \notin A$.

$\mathbb{S}_{cpf}$ will contain tiles over the following set of binding domains: $\{null, \mathsf{x}, \gamma, \mathsf{n}, \mathsf{e}, \mathsf{s}, \mathsf{w}\}$. Fig. 4 shows the start (S) and goal (G) tiles. The start tile has an n, e, s, and w binding domain on its north, east, south, and west sides, respectively, and as before, the goal tile is covered with γ binding domains.

Fig. 5 shows the four tiles of T_{cpf}. Each tile is labeled with an arrow (we explain the meaning of the arrow later). The glue strength function g_{cpf} is defined as follows:

- $\forall \sigma \in \Sigma_{cpf}, g_{cpf}(\sigma, null) = g_{cpf}(null, \sigma) = 0$,
- $\forall \sigma \in \Sigma_{cpf}, g_{cpf}(\sigma, \mathsf{x}) = g_{cpf}(\mathsf{x}, \sigma) = 0$,
- $\forall \sigma \in \Sigma_{cpf} \setminus \{null, \mathsf{x}\}, g_{cpf}(\sigma, \gamma) = g_{cpf}(\gamma, \sigma) = 1$,
- $\forall \sigma \in \Sigma_{cpf} \setminus \{null, \mathsf{x}, \gamma\}, g_{cpf}(\sigma, \sigma) = 2$,
- $\forall \sigma, \sigma' \in \Sigma_{cpf} \setminus \{\gamma\}$, if $\sigma \neq \sigma'$ then $g_{cpf}(\sigma, \sigma') = 0$.

Intuitively, the *null* and x binding domains have binding strength 0 to every other binding domain; the γ binding domain has binding strength 1 to every other binding domain, except *null* and x; and all other binding domains have binding strength 2 to themselves and 0 to others.

Let us examine the intuition behind $\mathbb{S}_{cpf} = \langle T_{cpf}, g_{cpf}, 2, 3 \rangle$. Fig. 6(a) shows a sample seed configuration with a four-tile obstacle. At temperature 2, tiles will attach to S to create possible paths toward G. Each of the four tiles in T_{cpf} is designed to attach in a specific way to an existing assembly: each of the tiles has exactly one of its four domains be "irregular" (where "regular" means n for north, e for east, s for south, and w for west). The growing assembly will always have regular domains on all its exposed sides, thus tiles may only attach via their single irregular domain. Fig. 6(b) and (c) show tile attachments after 1 and 2 steps, respectively. Paths may turn and fork in their attempts to reach G. Once G is reached, the last tile attaches with a total strength 3 (2 via its irregular binding domain and 1 to G). Fig. 6(d) shows some possible attachments after the system has been running for some time and has reached G. Once the temperature is increased to 3, all partial paths detach, while the successful paths remain. Fig. 6(e) shows the final configuration encoding a single path from S to G that avoids the obstacles.

To show that $\mathbb{S}_{cpf}$ solves the path-finding problem, we need a notion of distance in systems with obstacles. Formally, if T is a set of tiles containing *obstacle*, $A : \mathbb{Z} \to T$ is a configuration of T and $(x, y), (x', y') \in \mathbb{Z}^2$, then the **obstructed Manhattan distance** between (x, y) and (x', y') with respect to A is $n - 1$, for the least n such that there exists an untiled path $\langle p_1, p_2, \ldots, p_n \rangle$ of length n such that $p_1 = (x, y)$ and $p_n = (x', y')$ and for all $i \in \{1, 2, \ldots, n\}, A(p_i) \neq obstacle$.

That is to say, the obstructed Manhattan distance between two points on a 2D grid is the fewest number of unit-sized steps one has to take from one point to get to the other, without stepping on an obstacle. Fig. 7 shows an example grid with the obstructed Manhattan distances from the position marked 0. (If a point is not reachable from another because of obstacles, we say that the obstructed Manhattan distance between those points is ∞.) Note that the obstructed Manhattan distance could be far greater than the Manhattan distance. Fig. 8 shows an example obstacle arrangement in which the obstructed Manhattan distance is on the order of the Manhattan distance squared.

Theorem 2. *The melting tile system* $\mathbb{S}_{cpf} = \langle T_{cpf}, g_{cpf}, 2, 3 \rangle$ *solves the path-finding problem with obstacles.*

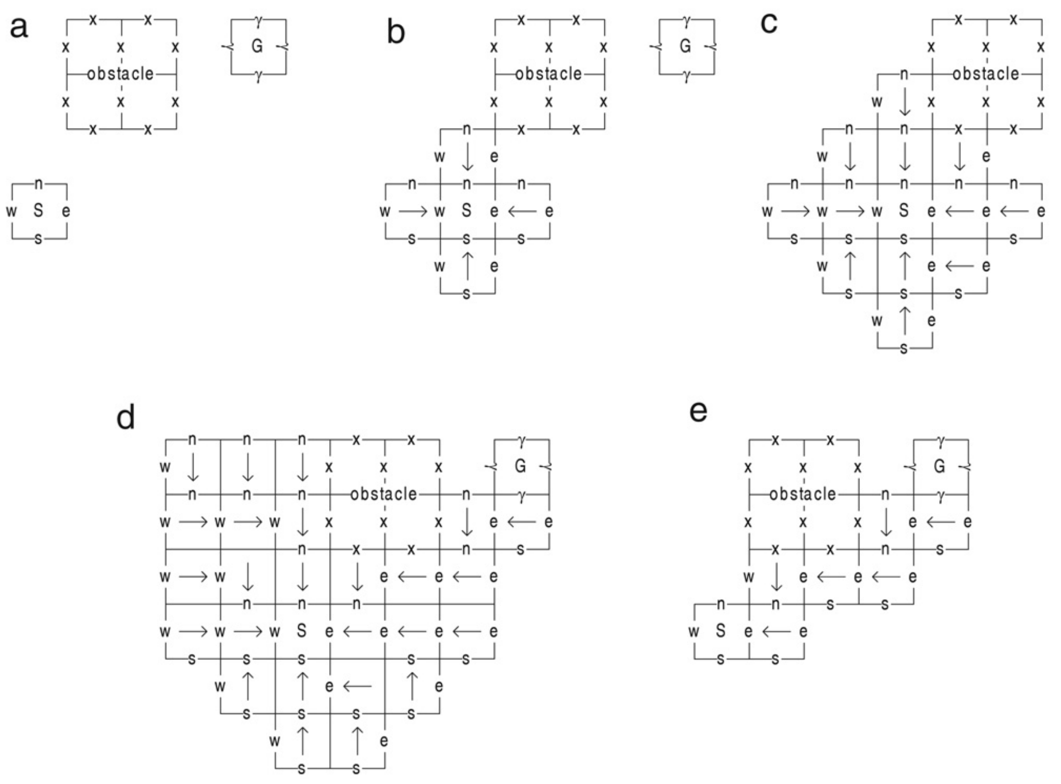

Fig. 6. An example execution of $\mathbb{S}_{cpf}$. $\mathbb{S}_{cpf}$ works at temperature 2 (a–d) to build possible paths and then at temperature 3 (e) to prune unsuccessful paths. Only binding domains that are exposed or attached have been labeled.

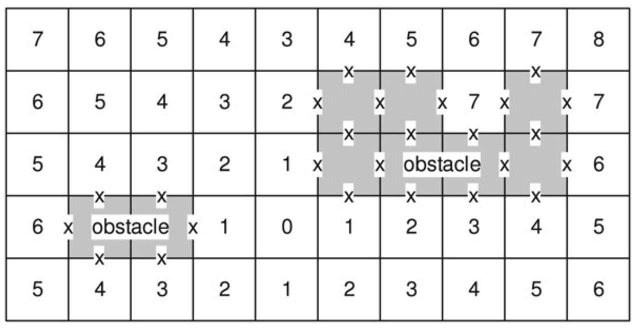

Fig. 7. The obstructed Manhattan distances on a 2D grid.

Proof. Let S_0 be a seed configuration such that $S_0(x_{\mathsf{S}}, y_{\mathsf{S}}) = \mathsf{S}$ and $S_0(x_{\mathsf{G}}, y_{\mathsf{G}}) = \mathsf{G}$ and for all other positions (x, y), $S_0(x, y) \in \{empty, obstacle\}$. Let d be the obstructed Manhattan distance from S to G. We will show by induction that after $t < d$ steps:

(1) All positions with obstructed Manhattan distance $\leq t$ from $(x_{\mathsf{S}}, y_{\mathsf{S}})$ will not be *empty*.
(2) All positions with obstructed Manhattan distance $>t$ from $(x_{\mathsf{S}}, y_{\mathsf{S}})$ will be *empty* (except for $(x_{\mathsf{G}}, y_{\mathsf{G}})$ and positions containing obstacle tiles).
(3) All exposed north binding domains will be n (except on G).
(4) All exposed south binding domains will be s (except on G).
(5) All exposed east binding domains will be e (except on G).
(6) All exposed west binding domains will be w (except on G).
(7) All tiles except S and G have an arrow that points to a position with a smaller obstructed Manhattan distance from $(x_{\mathsf{S}}, y_{\mathsf{S}})$, and will be attached to the tile in that position with strength 2.

For $t = 0$, all the conditions are trivially satisfied from S_0. Let S_t be a configuration that $\mathbb{S}_{cpf}$ produces on S_0 in $t < d$ steps. Assume that S_t satisfies conditions 1–7 above.

Note that at temperature 2, no tiles can attach to the *obstacle* tiles because their binding domains have 0-strength bonds to all other tiles, and no tiles will bind to G without binding to something else as well because its binding domains have 1-strength bonds to the four tiles in T_{cpf}. Further, if a tile from T_{cpf} can attach to G at temperature 2, it would also be able to attach in that position if G were not there, because all the binding domains on tiles from T_{cpf} attach to S and to other tiles from T_{cpf} with strength 2.

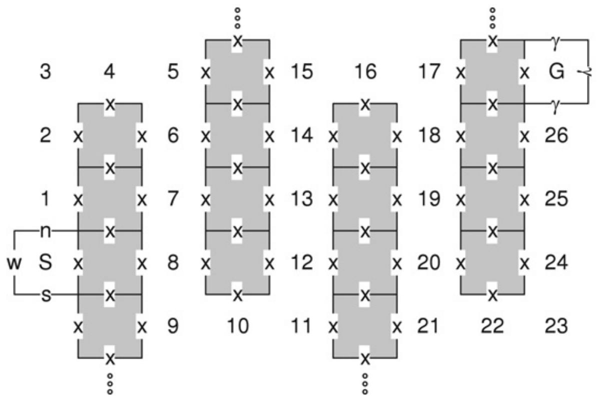

Fig. 8. A situation in which the obstructed Manhattan distance is on the order of the Manhattan distance squared. The obstructed Manhattan distance between S and G is 27, while the Manhattan distance is only 12. One can design obstacle formations to make the obstructed Manhattan distance arbitrarily greater than Manhattan distance, including infinite.

Similarly to the direct path-finding case, condition (2) and the facts that tiles may only attach in positions neighboring positions in S_t and no tile may attach to G without also attaching to at least one other tile in S_t, together imply that after step $t + 1$ no tile from T will be in a position with obstructed Manhattan distance $>t + 1$ from (x_S, y_S), so S_{t+1} will satisfy condition (2). By condition (1), every position (x, y) with obstructed Manhattan distance $t + 1$ from (x_S, y_S) will be adjacent to a position in S_t and by conditions (3), (4), (5), and (6) and by inspection of the tiles in T_{cpf}, at least one tile can attach in position (x, y). Hence, after step $t + 1$ every position with obstructed Manhattan distance $\le t + 1$ will not be *empty*, so S_{t+1} will satisfy condition (1).

Consider a position (x, y) with obstructed Manhattan distance $t+1$ from (x_S, y_S). By (1) and (2), $(x, y) \notin S_t$ and neighbors at least one position $(x', y') \in S_t$. If (x, y) is such that $(x, y) = N(x', y')$ (respectively $S(x', y')$, $E(x', y')$, $W(x', y')$) then $bd_N(S_t(x', y')) = \mathsf{n}$ (resp. $bd_S(S_t(x', y')) = \mathsf{s}$, $bd_E(S_t(x', y')) = \mathsf{e}$, $bd_W(S_t(x', y')) = \mathsf{w}$) by condition (3) (resp. (4), (5), (6)) so tile $\downarrow$ (resp. $\uparrow$, $\leftarrow$, $\rightarrow$) can attach in position (x, y) with strength 2. The arrow points toward position (x', y'), which has a smaller obstructed Manhattan distance from (x_S, y_S) than (x, y), so S_{t+1} satisfies condition (7). Finally, since tile $\downarrow$ (resp. $\uparrow$, $\leftarrow$, $\rightarrow$) attached to S_t via its south (resp. north, west, east) binding domain, the only possible exposed binding domains are on the other three sides, and by inspection of T_{cpf} all these binding domains satisfy conditions (3), (4), (5), and (6), so S_{t+1} will satisfy those conditions.

Note that there exists a path between S and G iff the obstructed Manhattan distance from S to G is finite. Thus if there exists a path, then after $d - 1$ steps, at least one tile t_{d-1} will have attached in a position p_{d-1} neighboring G by condition (1). Since t_{d-1} will be attached to some tile t_{d-2} in position p_{d-2} with strength 2 and to G with strength 1, t_{d-1} is attached with strength 3. Following the arrow on t_{d-1} leads to t_{d-2}. Then t_{d-2} is attached to two tiles (t_{d-1} and the tile in the direction of the arrow on t_{d-2}), and thus with strength 4; further, the obstructed Manhattan distance from p_{d-2} to S is smaller than from p_{d-1} to S. Repeating this process, following the arrows, we can construct a sequence of positions $\langle (x_S, y_S), p_1, p_2, \dots, p_{d-2}, p_{d-1}, (x_G, y_G) \rangle$. This sequence is a path in S_{d-1} of length $d + 1$ from S to G and each tile on the path (except S and G) is attached to other tiles on the path with strength at least 3.

Consider a tile t_1 in a position not on a minimum length path in S_{d-1}. Either t_1 has a tile t_2 in a neighboring position with an arrow pointing at t_1 or not. If not, then since t_1 is not on a minimum length path, it is not attached to G, so the only tile it can be attached to with positive strength is the tile in the direction of the arrow on t_1. This attachment is of strength 2, so t_1 will detach at temperature 3. If t_1 has a neighboring tile t_2 with an arrow pointing at t_1, then t_2 is also not on a minimum length path so the same argument applies. This process can continue for at most $d - 1$ steps at which point t_{d-1} must not have a neighboring tile pointing at it. Then t_{d-1} will detach at temperature 3, which will lead to t_{d-2} detaching in the next time step and so on.

Therefore at temperature 3, the only tiles that will remain in the final configuration are the tiles on a minimum length path. Thus given a seed configuration S_0, $\mathbb{S}_{cpf}$ produces a final configuration F on S_0 with switch times $t = d - 1$ and F encodes a path from S to G that avoids obstacles iff there exists at least one such path. □

Corollary 2. *Let S_0 be a seed configuration such that $S_0(x_S, y_S) = \mathsf{S}$ and $S_0(x_G, y_G) = \mathsf{G}$ and for all other positions (x, y), $S_0(x, y) \in \{empty, obstacle\}$. Let d be the obstructed Manhattan distance between (x_S, y_S) and (x_G, y_G). Then $\mathbb{S}_{cpf}$ produces a final configuration F on S_0 with switch time $d - 1$ and assembly time $\Theta(d)$. Further, for all $t \ge d - 1$, $\mathbb{S}_{cpf}$ produces a final configuration F on S_0 with switch time t, such that:*

(1) *There exists a path from* S *to* G *in F of length $d + 1$.*
(2) *For all $(x, y) \in F$, if $F(x, y) \neq obstacle$ then (x, y) is on a path from* S *to* G.

Proof. Part (1) follows from Theorem 2 and the fact that once a path of minimal length is assembled, each tile on the path is attached with strength at least 3, so will not detach from the path at temperature 3. The assembly time is the time required to attach tiles for form the path at temperature 2 plus the time required to detach all tiles that are not on a path from S to G. The assembly time is $2(d - 1) = \Theta(d)$.

To show part (2), let S_t be a configuration produced by $\mathbb{S}_{cpf}$ on S_0 with switch time $t \geq d - 1$ in t steps. If a tile t_1 is not on a path from S to G in S_t, then either there is no tile in a neighboring position with an arrow pointed at t_1 (in which case t_1 will detach in step $t + 1$ because the temperature will be 3), or every tile t_2 in a neighboring position that has an arrow pointing at t_1 is also not on a path from S to G. The same argument applies to t_2, so it follows that there must exist a tile t_k such that t_1 is on a path from S to t_k and t_k does not have a neighboring tile with an arrow pointing at t_k. Thus t_k will detach in step $t + 1$ and t_1 will detach in step $t + k$. Therefore the final configuration F will not have any tiles (except for *obstacle*) in positions not on a path from S to G. □

Note that we have been assuming maximum parallelism in our systems. That is, whenever a tile can attach, it does, and whenever a tile has to detach, it does. In physical implementations of the tile assembly model, it is far more likely that attachment and detachments happen stochastically, with rates that are related to the bond strength. While $\mathbb{S}_{dpf}$ always produces optimal-length paths, even without the maximum-parallelism assumption, $\mathbb{S}_{cpf}$ only produces optimal-length paths with the maximum-parallelism assumption. Theorem 2 does not hold if tiles may attach to positions with a greater obstructed Manhattan distance before attaching to ones with a lower obstructed Manhattan distance. One could analyze the expected length of the path in such an environment and find that with high probability, the length is on the order of the obstructed Manhattan distance.

In summary, $\mathbb{S}_{cpf}$ is guaranteed to find the shortest possible path between two points, in time linear in the length of the path, assuming maximum parallelism.

As before, while the theoretical definitions do not require knowing the proper switch time, in practice it may be helpful to be able to assess how long one has to wait before increasing the temperature. As Corollary 2 indicates, the switch time is $\Theta(d)$, where d is the obstructed Manhattan distance between S and G. More specifically, the proper minimum switch time is exactly $d - 1$. In practice, if the distance d is known, one can wait that long to increase the temperature. If d is unknown, one can devise an algorithm of increasing and decreasing the temperature repeatedly, perhaps increasing the length of switch time exponentially, such that in expectation a path can be found quickly.

5. Contributions

A number of biologically-inspired systems [1,8,15,18,25,27,30,37] attempt to use simple components with limited computational power to come together to accomplish complex tasks. The tasks commonly accomplished by these systems include finding paths between two points in 2D space and assembling shapes. The reason for the desire to use simple components is that they are cheap to produce in bulk.

We have presented two tile assembly systems, one that finds paths between two points in unrestricted space, and one that finds paths between two points even with obstacles present. These systems use components far simpler than those used in the related work. The components' interfaces are static and each component performs no computation, has no controlled movement, and has no writable memory (the binding domains could be considered read-only memory). Components such as these have been built out of DNA [9,16,23,33,35] at incredibly low costs.

While we have not gone into discussion of fault-tolerance within tile systems, a whole field of related research exists on making tile systems tolerant to individual tile failures [10,17,31,38,43,44]. One could apply the ideas in that related work directly to the tile systems we describe in this paper to make these systems able to perform successfully despite high probabilities of tiles failing, usually at the cost of slowing down the system assembly.

References

[1] H. Abelson, D. Allen, D. Coore, C. Hanson, G. Homsy, T.F. Knight Jr., R. Nagpal, E. Rauch, G.J. Sussman, R. Weiss, Amorphous computing, Communications of the ACM 43 (5) (2000) 74–82.
[2] L. Adleman, Towards a mathematical theory of self-assembly, Tech. Rep. 00722, Department of Computer Science, University of Southern California, Los Angleles, CA, 2000.
[3] L. Adleman, Q. Cheng, A. Goel, M.-D. Huang, D. Kempe, P. Moisset de Espanés, P.W.K. Rothemund, Combinatorial optimization problems in self-assembly, in: Proceedings of the 34th Annual ACM Symposium on Theory of Computing, STOC02, Montreal, Quebec, Canada, 2002.
[4] L. Adleman, Q. Cheng, A. Goel, M.-D. Huang, H. Wasserman, Linear self-assemblies: Equilibria, entropy, and convergence rates, in: Proceedings of the 6th International Conference on Difference Equations and Applications, ICDEA01, Augsburg, Germany, 2001.
[5] L. Adleman, A. Goel, M.-D. Huang, P. Moisset de Espanés, Running time and program size for self-assembled squares, in: Proceedings of the 34th Annual ACM Symposium on Theory of Computing, STOC02, Montreal, Quebec, Canada, 2002.
[6] L. Adleman, J. Kari, L. Kari, D. Reishus, On the decidability of self-assembly of infinite ribbons, in: Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science, FOCS02, Ottawa, Ontario, Canada, 2002.
[7] G. Aggarwal, Q. Cheng, M.H. Goldwasser, M.-Y. Kao, P. Moisset de Espanés, R.T. Schweller, Complexities for generalized models of self-assembly, SIAM Journal on Computing 34 (6) (2005) 1493–1515.
[8] D.J. Arbuckle, A.A.G. Requicha, Active self-assembly, in: Proceedings of the IEEE International Conference on Robotics and Automation, ICRA04, New Orleans, LA, USA, 2004.
[9] R. Barish, P.W.K. Rothemund, E. Winfree, Two computational primitives for algorithmic self-assembly: Copying and counting, Nano Letters 5 (12) (2005) 2586–2592.
[10] Y. Baryshnikov, E.G. Coffman, N. Seeman, T. Yimwadsana, Self correcting self assembly: Growth models and the hammersley process, in: Proceedings of the 11th International Meeting on DNA Computing, DNA05, London, Ontario, 2005.
[11] R. Berger, The Undecidability of the Domino Problem, in: Memoirs Series, vol. 66, American Mathematical Society, 1966.
[12] Y. Brun, Arithmetic computation in the tile assembly model: Addition and multiplication, Theoretical Computer Science 378 (1) (2007) 17–31.
[13] Y. Brun, Nondeterministic polynomial time factoring in the tile assembly model, Theoretical Computer Science 395 (1) (2008) 3–23.
[14] Y. Brun, Solving NP-complete problems in the tile assembly model, Theoretical Computer Science 395 (1) (2008) 31–46.

[15] W.J. Butera, Programming a paintable computer, Ph.D. Thesis, Massachussetts Institute of Technology, Cambridge, MA, USA, February 2002.
[16] N. Chelyapov, Y. Brun, M. Gopalkrishnan, D. Reishus, B. Shaw, L. Adleman, DNA triangles and self-assembled hexagonal tilings, Journal of American Chemical Society (JACS) 126 (43) (2004) 13924–13925.
[17] H.-L. Chen, A. Goel, Error free self-assembly with error prone tiles, in: Proceedings of the 10th International Meeting on DNA Based Computers, DNA04, Milan, Italy, 2004.
[18] L. Clement, R. Nagpal, Self-assembly and self-repairing topologies, in: Proceedings of the Workshop on Adaptability in Multi-Agent Systems, RoboCup Australian Open, 2003.
[19] M. Cook, P.W.K. Rothemund, E. Winfree, Self-assembled circuit patterns, in: Proceedings of the 9th International Meeting on DNA Based Computers, DNA03, Madison, WI, USA, 2003.
[20] H.-D. Ebbinghaus, Undecidability of some domino connectability problems, Zeitschrift für Mathematische Logik und Grundlagen der Mathematik 28 (1982) 331–336.
[21] H.-D. Ebbinghaus, Domino threads and complexity, Lecture Notes in Computer Science (1987) 131–142.
[22] Y. Etzion-Petruschka, D. Harel, D. Myers, On the solvability of domino snake problems, Theoretical Computer Science 131 (2) (1994) 243–269.
[23] T.J. Fu, N.C. Seeman, DNA double-crossover molecules, Biochemistry 32 (13) (1993) 3211–3220.
[24] M.-Y. Kao, R. Schweller, Reducing tile complexity for self-assembly through temperature programming, in: Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA06, Miami, FL, USA, 2006.
[25] A. Kondacs, Biologically-inspired self-assembly of two-dimensional shapes using global-to-local compilation, in: Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI03, Acapulco, Mexico, 2003.
[26] M.G. Lagoudakis, T.H. LaBean, 2D DNA self-assembly for satisfiability, DIMACS Series in Discrete Mathematics and Theoretical Computer Science 54 (1999) 141–154.
[27] J. McLurkin, J. Smith, J. Frankel, D. Sotkowitz, D. Blau, B. Schmidt, Speaking swarmish: Human-robot interface design for large swarms of autonomous mobile robots, in: Proceedings of the AAAI Spring Symposium, Stanford, CA, USA, 2006.
[28] P. Moisset de Espanés, Computerized exhaustive search for optimal self-assembly counters, in: Proceedings of the 2nd Foundations of Nanoscience: Self-Assembled Architectures and Devices, FNANO05, Snowbird, UT, USA, 2005.
[29] D. Myers, Decidability of the tiling connectivity problem, Notices of the American Mathematical Society 195 (26) (1979) A-441.
[30] R. Nagpal, Programmable self-assembly: Constructing global shape using biologically-inspired local interactions and origami mathematics, Ph.D. Thesis, Massachussetts Institute of Technology, Cambridge, MA, USA, June 2001.
[31] J.H. Reif, S. Sahu, P. Yin, Compact error-resilient computational DNA tiling assemblies, in: Proceedings of the 10th International Meeting on DNA Based Computers, DNA04, Milan, Italy, 2004.
[32] D. Reishus, Design of a self-assembled memory circuit, in: Proceedings of the 5th Foundations of Nanoscience: Self-Assembled Architectures and Devices, FNANO08, Snowbird, UT, USA, 2008.
[33] D. Reishus, B. Shaw, Y. Brun, N. Chelyapov, L. Adleman, Self-assembly of DNA double-double crossover complexes into high-density, doubly connected, planar structures, Journal of American Chemical Society (JACS) 127 (50) (2005) 17590–17591.
[34] R.M. Robinson, Undecidability and nonperiodicity for tilings of the plane, Inventiones Mathematicae 12 (3) (1971) 177–209.
[35] P.W.K. Rothemund, N. Papadakis, E. Winfree, Algorithmic self-assembly of DNA Sierpinski triangles, PLoS Biology 2 (12) (2004) e424.
[36] P.W.K. Rothemund, E. Winfree, The program-size complexity of self-assembled squares, in: Proceedings of the 32nd Annual ACM Symposium on Theory of Computing, STOC00, Portland, OR, USA, 2000.
[37] W.-M. Shen, M. Krivokon, H. Chiu, J. Everist, M. Rubenstein, J. Venkatesh, Multimode locomotion for reconfigurable robots, Autonomous Robots 20 (2) (2006) 165–177.
[38] D. Soloveichik, M. Cook, E. Winfree, Combining self-healing and proofreading in self-assembly, Natural Computing 7 (2) (2008) 203–218.
[39] H. Wang, Proving theorems by pattern recognition, II, Bell System Technical Journal 40 (1961) 1–42.
[40] E. Winfree, On the computational power of DNA annealing and ligation, DNA Based Computers (1996) 199–221.
[41] E. Winfree, Algorithmic self-assembly of DNA, Ph.D. Thesis, California Institute of Technology, Pasadena, CA, USA, June 1998.
[42] E. Winfree, Simulations of computing by self-assembly of DNA, Tech. Rep. CSTR:1998:22, California Institute of Technology, Pasadena, CA, USA, 1998.
[43] E. Winfree, Self-healing tile sets, Nanotechnology: Science and Computation (2006) 55–78.
[44] E. Winfree, R. Bekbolatov, Proofreading tile sets: Error correction for algorithmic self-assembly, in: Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science, FOCS02, vol. 2943, Madison, WI, USA, 2003.
[45] H. Yan, S.H. Park, G. Finkelstein, J.H. Reif, T.H. LaBean, DNA-templated self-assembly of protein arrays and highly conductive nanowires, Science 301 (2003) 1882–1884.