



Drainability and Fillability of Polyominoes in Diverse Models of Global Control

Sándor P. Fekete ✉ 

Computer Science, TU Braunschweig, Germany

Peter Kramer ✉ 

Computer Science, TU Braunschweig, Germany

Jan-Marc Reinhardt ✉ 

Electrical Engineering and Computer Science, Bochum University of Applied Sciences, Germany

Christian Rieck ✉ 

Discrete Mathematics, University of Kassel, Germany

Christian Scheffer ✉ 

Electrical Engineering and Computer Science, Bochum University of Applied Sciences, Germany

Abstract

Tilt models offer intuitive and clean definitions of complex systems in which particles are influenced by global control commands. Despite a wide range of applications, there has been almost no theoretical investigation into the associated issues of filling and draining geometric environments. This is partly because a globally controlled system (i.e., passive matter) exhibits highly complex behavior that cannot be locally restricted. Thus, there is a strong need for theoretical studies that investigate these models both (1) in terms of relative power to each other, and (2) from a complexity theory perspective. In this work, we provide (1) general tools for comparing and contrasting different models of global control, and (2) both complexity and algorithmic results on filling and draining.

2012 ACM Subject Classification Theory of computation → Computational geometry

Keywords and phrases Global control, full tilt, single tilt, fillability, drainability, polyominoes, complexity

Funding Work from TU Braunschweig and HS Bochum was partially supported by the German Research Foundation (DFG), project “Space Ants”, FE 407/22-1 and SCHE 1931/4-1. Work from University of Kassel was partially supported by DFG grant 522790373.

Acknowledgements We appreciate the detailed feedback provided by the anonymous reviewers. We thank Erik D. Demaine for early helpful conversations.

1 Introduction

The targeted use of global control mechanisms, applied synchronously and uniformly¹ to small-particle matter (i.e., *passive matter*), is both of great practical relevance and highest (theoretical) complexity. A fundamental type of global control mechanism is that of uniform movement or translation, which finds application in a variety of manufacturing processes, such as filling polyhedra with a liquid for gravity casting [19, 20, 30] or the intact removal of cast objects from their molds [17, 18]. Global forces like electromagnetic fields and gravity play a key role in a range of further applications, such as amorphous computing and smart materials like smart paint [2], autonomous monitoring and repair systems as well as minimally invasive surgeries [11], medication [14], and biological robots [16].

¹ Due to this intrinsic connection, we use the qualifiers “uniform” and “global” synonymously when applied to models of motion.

Inspired by tilt games [1, 4, 24], theoretical research commonly uses discrete, grid-based models to study the manipulation of passive matter by global control signals. In this context, a distinction is made between single step and full tilt models [11, 23], i.e., the movement of all particles by either one step or by a maximum distance in a uniform direction.

One of the most challenging problems remains the question of filling geometric shapes using global control: Given a *board*, defined as a subset of the square grid as well as a number of “infill points” (*sources*), the question is whether, and if so how, the entire board can be filled by adding particles through the set of sources, with all particles moving in the same direction as determined by global control mechanisms. Crucially, the inherently discrete nature of particle models introduces new complications that have not appeared in the continuous frameworks in [19, 20], such as particles forming stacks by blocking each other instead of spreading like liquids.

Naturally, there exist boards that cannot be filled with a given number of sources. We investigate the (therefore immediately arising) question of minimal necessary changes to the board in order to achieve fillability. Even in simplified models, globally controlled particles and targeted changes to the board create dynamic systems that are extremely complex to analyze and control: The smallest manipulation of a board can lead to far-reaching and not locally restrictable changes in terms of fillability, see for example Figures 2(b) and 2(c).

Furthermore, it is unclear how the different models relate to each other in terms of the contrasting objectives of filling or draining. In this paper, we provide a generalized, comprehensive framework for the comparative study of tilt models and formally investigate what makes a given model *more powerful* than another, i.e., allows more polyominoes to be filled or drained. This includes the special case of single step and full tilt movement.

1.1 Our Contributions

We build upon established models of particle movement using global control signals, namely the full tilt and single step models, examine them in the context of generalized and more powerful models, and investigate how to achieve drainability and fillability of polyominoes in these models by placing obstacles. Our main contributions are twofold.

- (1) We provide general tools to compare and contrast various models of particle movement, offering a more unified perspective and surprising new insights concerning the duality of different models (see Figure 1). In particular, we prove the following:
 - (1.1) Equivalence between drainability in the full tilt model and fillability in the single step model (Corollary 41).
 - (2.2) Limited relaxation of the restriction to global control signals does not tangibly affect drainability (Theorem 40), i.e., does not increase model power in this regard.
- (2) With regards to gaining drainability via obstacles in the full tilt model, we provide:
 - (2.1) A reduction from a 3SAT variant showing that it is NP-hard to decide whether a given number of obstacles suffices, even when restricted to thin polyominoes, i.e., those containing no 2×2 squares (Theorem 6).
 - (2.2) A c -approximation algorithm for k -scaled boards of $c = 4$ for $k = 3$ and of $c = 6$ for $k > 3$ (Theorem 21 and Corollary 22).

For ease of exposition, we first present algorithmic results for drainability in the full tilt model in Sections 2–4, before generalizing to fillability and other models in Section 5.



■ **Figure 1** Movement of a single bubble in the single step model (left) is identical to the movement of a single particle in the full tilt model (right).

1.2 Related Work

The drainability and fillability of polygonal two-dimensional shapes by uniform tilt movement has previously been studied by Aloupis et al. [6], who gave an exact polynomial-time algorithm for the minimum number of sinks (i.e., exits) necessary to drain a polygon or polyhedron. We study the discrete “tilt” model first introduced by Becker et al. [15, 23] in 2013. Particles on a grid-based *board* are moved uniformly, in either the single step or full tilt model:

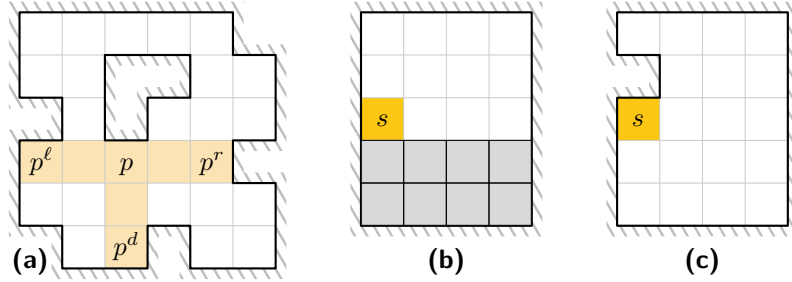
The *reconfiguration* problem asks whether a given arrangement of particles can be reconfigured into another specific arrangement, given a fixed set of obstacles. The minimization variant, i.e., finding a shortest sequence of moves, was shown to be PSPACE-complete in the full tilt model by Becker et al. [11, 12]. A natural subproblem is the *relocation* problem, which asks for just one specific particle to be moved to a target position. In this variant, even deciding existence of any movement sequence is NP-hard in the full tilt model [10, 11]. Recent complexity results for single step tilt moves by Caballero et al. demonstrate that, when restricted to two or three cardinal directions, deciding the existence of a relocation sequence is NP-complete [21], as well as in the absence of obstacles (i.e., a rectangular board) and just two movement directions [23]. The more general *occupancy* problem asks whether an arrangement of particles can be modified to move *any* particle to a specific target position. Even with this relaxation, finding a shortest sequence of moves, or deciding that no such sequence exists, is PSPACE-complete [8, 11]. Caballero et al. [22] show that deciding occupancy remains PSPACE-complete in the single step model when 2×1 dominoes are considered in addition to 1×1 particles. However, it is easy to see that the problem is in P if only unit-size particles are considered.

A myriad of related problems have been studied in the tilt model, such as *gathering* particles into a connected configuration [14, 26], or the design of special-purpose boards for efficient reconfiguration: In particular, Balanza-Martinez et al. [9] studied the design of universal shape constructors, i.e., boards that can be used to create large classes of particle configurations. Further results exist on the reordering of labeled rectangular arrangements, which can be achieved either in linear time using quadratic area [11], or in quadratic time and linear area [31]. In addition to “workspace”-based tilt reconfiguration settings, a variety of other models exist, e.g., moving only particles at maximal coordinates [3, 5], rather than all particles. A number of questions remain on the classification and complexity of deciding which configurations can be constructed [13, 25] by sequentially introducing particles into an unobstructed system and “gluing” them onto an existing configuration using tilt movement.

2 Preliminaries

Particles move on a *board*, which we model as a finite subgraph of the square tiling’s dual graph, i.e., a board $B = (V, E)$ is a finite graph with $V \subset \mathbb{Z}^2$ and $E \subseteq \{\{u, v\} : u \in V, v \in V, \|u - v\|_1 = 1\}$. The *boundary* of a board is the axis-aligned polygon separating the tiles in V from $\mathbb{Z}^2 \setminus V$. Note that the boundary uniquely determines the corresponding board. We only illustrate the special case of vertex-induced boards, which are isomorphic to sets of polyominoes and as such have boundaries whose sides only intersect at corners, even

though the results in this paper hold in general. A *sub-board* is a vertex-induced subgraph of a board; a maximally connected (sub-)board is called a *region*. We refer to the elements of V as *pixels*. Every pixel is uniquely determined as the *intersection* of two *segments*, a horizontal *row segment* and a vertical *column segment*, which are maximally contiguous subsets of pixels of equal y -, or x -coordinate, respectively. A *boundary pixel* is a pixel adjacent to a side of the boundary; a *corner pixel* is a pixel adjacent to two perpendicular sides of the boundary. The set $\{\ell, r, u, d\}$ is shorthand for the directions left, right, up, and down. For a pixel p let p^ℓ , p^r , p^u , and p^d denote the left and right boundary pixel in its row segment, and the top and bottom boundary pixel in its column segment, respectively; see Figure 2(a).



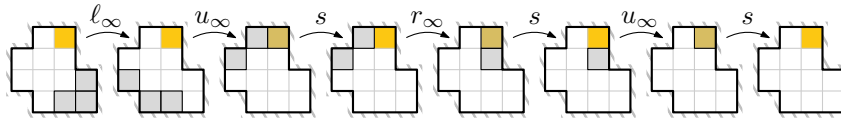
■ **Figure 2** (a) A board with a pixel p at the intersection of its row and column segments (shaded area), along with the boundary pixels of these segments. Note that $p = p^u$. (b) A configuration of a board B that is minimal with respect to a sink s in the full tilt model. Particles are depicted as gray squares. (c) A drainable sub-board of B .

A *configuration* $C \subseteq V$ is a subset of the pixels. We call a pixel $p \in C$ *occupied* and a pixel $p \in V \setminus C$ *free*. A *move* $m : 2^V \rightarrow 2^V$ is a mapping between configurations; a *model* is a set of moves. Given a model M and two configurations, C and D , we say D is *reachable from* C in one move, or $C \rightarrow_M D$, if $D = m(C)$ for some $m \in M$. D is *reachable from* C , or $C \rightarrow_M^* D$, if there is a sequence of moves $m_1, m_2, \dots, m_k \in M$ such that $D = m_k \circ \dots \circ m_2 \circ m_1(C)$, i.e., \rightarrow_M^* is the reflexive, transitive closure of \rightarrow_M . We omit braces for singletons and write, e.g., $u \rightarrow_M v$ instead of $\{u\} \rightarrow_M \{v\}$. A pixel p is *occupiable* from a configuration C if there is a configuration D reachable from C that contains p .

The *full tilt model*, $\text{FT} = \{u_\infty, d_\infty, \ell_\infty, r_\infty\}$, has one move for every direction, which sees particles move maximally in that direction until they hit the boundary or are blocked by another particle. The move ℓ_∞ , for example, transforms a configuration C in such a way that exactly the $|R \cap C|$ leftmost pixels of every row segment R are occupied in $\ell_\infty(C)$.

A *sink* is a move associated with a pixel $s \in V$ defined as $C \mapsto C \setminus \{s\}$. A configuration C is *minimal* with respect to a set of sinks $S \subseteq V$ if there is no configuration D such that $C \rightarrow_{\text{FT} \cup S}^* D$ and $|D| < |C|$. If $V \rightarrow_{\text{FT} \cup S}^* \emptyset$, then the board is *drainable* to S . Figure 3 illustrates particles moved to and removed at a sink.

Further notation is introduced as needed in individual sections. Table 1 provides a reference of frequently used symbols.



■ **Figure 3** Draining three particles to a sink s using full tilt moves.

■ **Table 1** Frequently used notation.

Notation	Meaning	Defined in
$C \rightarrow_M^* D$	D is reachable from C in model M	Section 2
$\text{FT} = \{u_\infty, d_\infty, \ell_\infty, r_\infty\}$	the full tilt model	Section 2
$G_F(B)$	the full tilt graph of a board B	Section 3
$G_S(B)$	the small tilt graph of a board B	Section 3
$B^{\uparrow k}$	the board B scaled by a factor k	Section 4.2
$G^{\leftrightarrow S}$	the extended graph of G with a set of sinks S	Section 4.2
$G_L(B, S)$	the large tilt graph of a board B and a set of sinks S	Section 4.2
$\text{S1} = \{u_1, d_1, \ell_1, r_1\}$	the single step model	Section 5
M_I	the interval extension of model $M \in \{\text{FT}, \text{S1}\}$	Section 5
\overline{M}	the dual model of M	Section 5.2

3 Drainability in the Full Tilt Model

In this section, we examine an algorithmic approach to deciding drainability in the full tilt model, starting with the following theorem. This is a special case of a property that holds in more general classes of models, which is why we postpone the proof until Theorem 29 in Section 5.

► **Theorem 1.** *A board $B = (V, E)$ is drainable to a set of sinks $S \subseteq V$ if and only if for every $p \in V$ there exists some $s \in S$ such that $p \rightarrow_{\text{FT}}^* s$.*

The characterization in Theorem 1 suggests an approach to deciding drainability. Consider what we call the *full tilt graph* $G_F(B) = (V_F, E_F)$ of a board $B = (V, E)$. $V_F = V$ contains all pixels of B and $E_F = \{(p, q) \in V^2 : p \neq q, q \in \{p^\ell, p^r, p^u, p^d\}\}$ connects p to q if a particle at p can reach q in a single move. Deciding drainability is equivalent to testing if there is a path in G_F from every pixel to a sink. However, this is unnecessarily inefficient. In particular, we do not need to check every pixel and require only the boundary as input.

► **Lemma 2.** *For every pixel p on a board there is a corner pixel c such that $p \rightarrow_{\text{FT}}^* c$.*

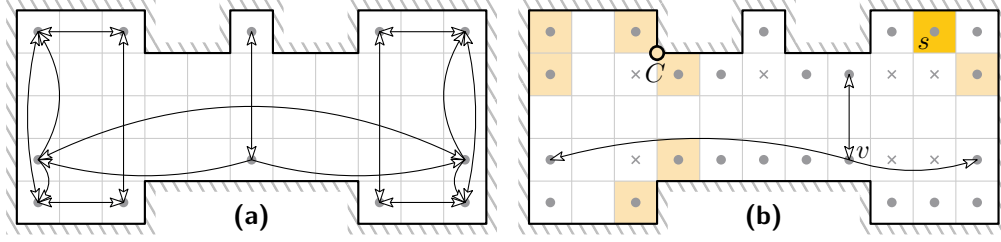
Proof. TOPROVE 0 ◀

► **Corollary 3.** *A board $B = (V, E)$ is drainable to a set of sinks $S \subseteq V$ if and only if for every corner pixel $c \in V$ there exists some $s \in S$ such that $c \rightarrow_{\text{FT}}^* s$.*

Proof. TOPROVE 1 ▶

We now describe an algorithm that, given the boundary of a board $B = (V, E)$ and a set $S \subseteq V$ of potential sinks, finds a minimum-cardinality subset $S' \subseteq S$ such that B is drainable to S' —or reports that no such set exists. As a consequence, we can decide whether a board is drainable to a set S by supplying S to the algorithm and observing if it returns a subset or reports failure. The general approach is the same as the one used by the authors of [6]. However, due to the rectilinear nature of board boundaries, and our restriction to tilting in only four directions, our algorithm requires less time.

By Corollary 3, the only pixels relevant for drainability are those reachable from corner pixels. This allows us to restrict our algorithm to the *small tilt graph*, $G_S(B) = (V_S, E_S)$, of B . Let $V' \subseteq V$ be the set of corner pixels of B . Then V_S is the closure of V' under $\rightarrow_{\text{FT}}^*$, and G_S is the subgraph of G_F induced by V_S ; see Figure 4(a). For a boundary with n



■ **Figure 4** (a) The small tilt graph, $G_S(B)$, of a board B . (b) The vertices of the large tilt graph, $G_L(B, \{s\})$, of B and a sink s . The lightly-shaded pixels are added due to the reflex corner C (though some are also part of the graph as corner pixels). Vertices marked with a cross are added as intersections. We only depict the edges incident on vertex v to avoid clutter.

corners, $|V_S| \in \mathcal{O}(n)$ as there are at most two additional pixels for every reflex corner of the boundary, and $|E_S| \in \mathcal{O}(n)$ as at most three other pixels are reachable from every boundary pixel in a single move.

► **Lemma 4.** $G_S(B)$ can be constructed from the boundary of a board B in $\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space, where n is the number of corners of the boundary.

Proof. TOPROVE 2 ◀

The next step is to consider the condensation $G_S^* = (V_S^*, E_S^*)$ of G_S , i.e., the directed graph that has as vertices the strongly-connected components of G_S and an edge from component A to component B if there is an edge from a vertex in A to a vertex in B in G_S . Then, B is drainable to every set S that contains at least one pixel from every sink $V' \in V_S^*$. A suitable subset of sinks, if one exists, can thus be computed in $\mathcal{O}(|V_S|)$ time and space. See Algorithm 1 for an overview of the steps.

■ **Algorithm 1** Finding a minimum-cardinality set of sinks.

Input: The boundary of a board $B = (V, E)$ and a set $S \subseteq V$.
Output: A minimum-cardinality subset $S' \subseteq S$ such that $V \rightarrow_{\text{FTUS}'}^* \emptyset$, or a message indicating no such set exists.

- 1 Construct G_S using Lemma 4.
- 2 Compute $G_S^* = (V_S^*, E_S^*)$, the condensation of G_S .
- 3 $S' := \emptyset$
- 4 **foreach** sink $V' \in V_S^*$ **do**
- 5 **if** there is at least one $s \in V' \cap S$ **then**
- 6 $S' := S' \cup \{s\}$
- 7 **else**
- 8 **return** No solution exists because $V' \cap S = \emptyset$.
- 9 **end**
- 10 **end**
- 11 **return** S'

► **Theorem 5.** Algorithm 1 produces a correct result and runs in $\mathcal{O}(n \log n + |S|)$ time using $\mathcal{O}(n)$ space, where n is the number of corners of the boundary.

Proof. TOPROVE 3 ◀

4 Placing Obstacles to Guarantee Drainability

Although not every board is drainable, slight changes may render a board drainable, see Figures 2(b) and 2(c). Given a board and a set S of sinks, we want to determine a maximum-size sub-board drainable to S . In other words, we want to place a minimum number of *obstacles* such that the remaining board is drainable. We refer to this as the MAXIMUM TILT DRAINING PROBLEM.

4.1 Computational Complexity

In this section, we show that the problem is NP-hard; in particular, we show that deciding whether ℓ many obstacles suffice to guarantee that a board is drainable is an NP-hard problem. Our reduction is from 3SAT-3 [29], and works as follows; we refer to Figure 5 for an overview.

The problem 3SAT-3 is a variant of 3SAT having the additional property that every variable appears at most 3 times; thus, we may assume that each variable occurs at least once negated, and once unnegated. Then, for every instance φ of 3SAT-3, we construct a polyomino P_φ that is an instance of the MAXIMUM TILT DRAINING PROBLEM.

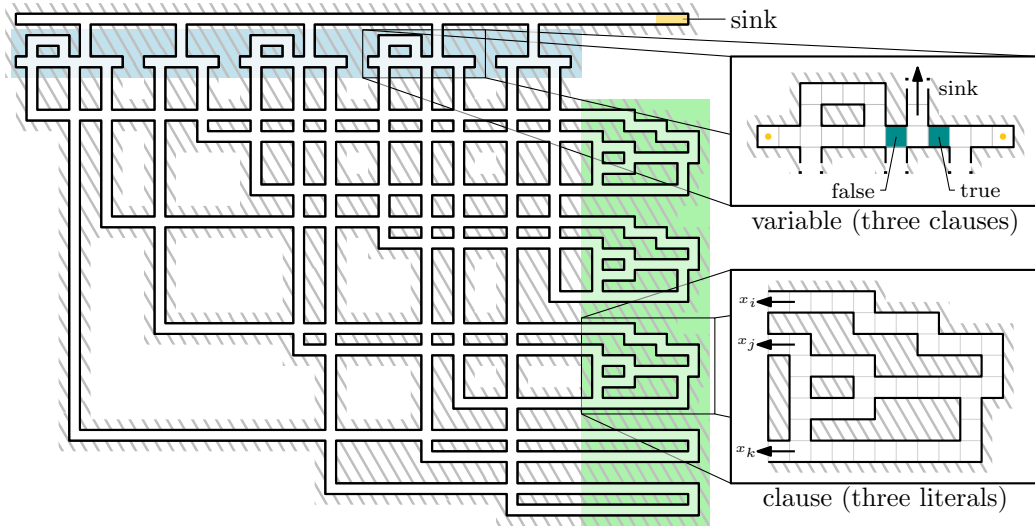


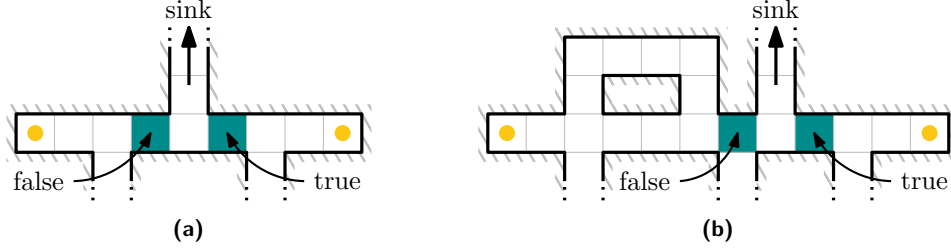
Figure 5 Overview of the NP-hardness reduction for the full tilt variant. The depicted instance is due to the 3SAT-3 formula $\varphi = (x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_4 \vee \neg x_5) \wedge (x_2 \vee x_3 \vee \neg x_4) \wedge (x_1 \vee x_4) \wedge (\neg x_3 \vee x_5)$.

For this, we add for each variable the respective variable gadget. As illustrated in Figure 5, the variable gadgets are placed in row at the top of the polyomino. For each clause, we place the respective clause gadget vertically in row at the right side of the construction. We then connect each variable with its respective clauses by thin L-shaped corridors.

It is easy to observe that we need at least one obstacle per variable to drain a variable gadget of P_φ . Furthermore, from every other position we can reach a variable gadget, hence, no further obstacles are needed. By carefully arguing we can show that P_φ is drainable if and only if ℓ many obstacles are placed at very specific locations within the variable gadgets, where ℓ is the number of variables in φ . This leads to the following theorem.

► **Theorem 6.** *It is NP-hard to decide whether placing k obstacles suffices to guarantee drainability of a thin polyomino.*

We specify variable and clause gadgets, and how these are connected to one another, and start with the variable gadget as depicted in Figure 6.



■ **Figure 6** (a) Variable gadget for two occurrences, and (b) for three occurrences.

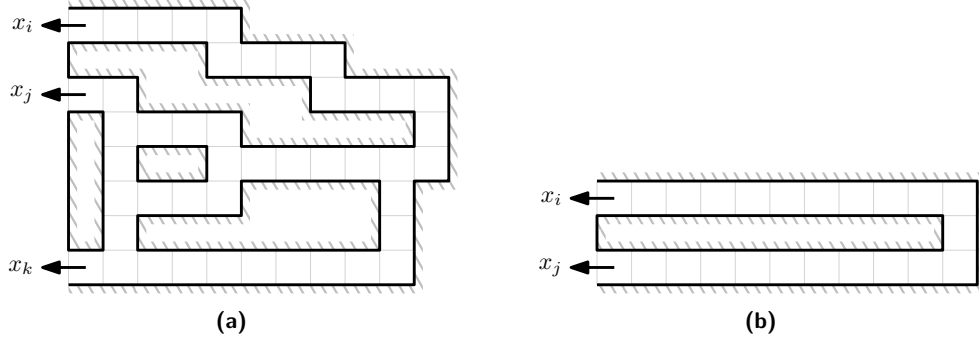
► **Lemma 7.** *To drain the variable gadget, we need to place at least one obstacle. Moreover, placing one obstacle on one of two pixels within the gadget is sufficient to drain it, if their entrances are connected by a path.*

Proof. TOPROVE 4 ◀

As there are exactly two options for placing an obstacle in the desired manner within a variable gadget, we exploit these options for distinguishing between true and false. Combining Theorem 1 and Lemma 7 directly implies that an obstacle, placed somewhere outside of a variable gadget, has no influence on whether a particle can leave that variable gadget or not. Additionally, as all variables are disjoint in our construction, we observe the following.

► **Corollary 8.** *For every variable gadget, we need a unique obstacle to drain it.*

We proceed with the clause gadget, as depicted in Figure 7.



■ **Figure 7** (a) shows a clause gadget that contains three literals, while (b) contains two literals.

► **Lemma 9.** *To reach the respectively contained variable gadgets from every corner pixel of a clause, we do not need to place any obstacles.*

Proof. TOPROVE 5 ◀

By construction, a particle leaves a clause gadget horizontally, and reaches a convex corner pixel of the connecting path to the respective variable, from which the particle can either move back (and reaches a convex corner pixel of the clause gadget we came from), or it moves vertically into a variable gadget. From there, we either move back, or horizontally;

by the first alternative, the particle cannot reach any new pixels, and by the second it is trapped within the variable gadget, if there is no already placed obstacle. Thus, we make the following observation.

► **Observation 10.** *Without placing any obstacles, a particle on any pixel of a clause gadget cannot move to any pixel of another clause gadget by any tilt sequence. The placement of obstacles within a variable also imply that a clause is only path-connected to another clause when they both contain the same literal.*

With this, we can prove Theorem 6, which we restate here.

► **Theorem 6.** *It is NP-hard to decide whether placing k obstacles suffices to guarantee drainability of a thin polyomino.*

Proof. TOPROVE 6 ◀

4.2 An Approximation Algorithm for Scaled Boards

Our hardness proof relies on thin segments in the variable gadgets. To achieve positive results, we restrict the considered boards to ones that do not have thin segments by introducing scaling. In a k -scaled board $B^{\uparrow k}$ of a board B , every tile of the underlying square tiling gets replaced by a $k \times k$ grid of tiles. This subdivides every pixel p into k^2 sub-pixels, for which p is their super-pixel. Sinks get scaled as well, which means that the move associated with a sink may remove up to k^2 particles at once. By Corollary 3, only the sub-pixels reachable from corner sub-pixels are relevant in terms of drainability. We call these *outer* sub-pixels; the remaining sub-pixels are *inner* sub-pixels. Without any placed obstacles, only sub-pixels at a corner of their super-pixel can possibly be outer sub-pixels, see Figure 8(d).

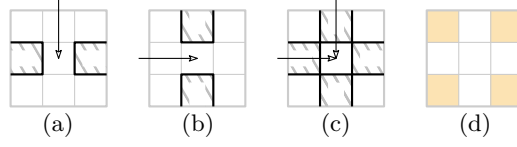
The basic idea of our approach is to employ k -scaling, for $k \geq 3$, and use inner sub-pixels as positions for obstacles, leading to new outer sub-pixels as intermediate steps on paths from every pixel to a sink. We assume that a given board contains a sink in every region—the only way to handle a region without a sink is to fill it with obstacles. The chief benefit we get from obstacles are new edges in the full tilt graph. We want as few new edges as possible while guaranteeing the existence of paths from every corner pixel to a sink.

For a directed graph $G = (V, E)$ and a set of sinks $S \subseteq V$, we define the *extended graph* $G^{\leftrightarrow S} = (V \cup \{r\}, E^{\leftrightarrow S})$ to be the weighted super-graph of G that contains all edges $e \in E$ with weight zero and, for every edge $(p, q) \in E$ with $(q, p) \notin E$, the *inverse edge* (q, p) with weight one, in addition to edges (s, r) with weight zero from every sink s to a newly added vertex r , called the *root*. Note that the extended graph contains an arborescence converging to r , as long as there is a sink in every undirected component of the original graph.

Conceptually, we place *turn gadgets* at sub-pixels of the heads of inverse edges in a minimum-weight arborescence of the full tilt graph. For scaling factor 3, a turn gadget consists of two obstacles placed perpendicular to the direction of the inverse edge in the middle of the pixel, see Figures 8(a) to (c). The central sub-pixel in Figure 8(c) gets disconnected by the obstacles from the rest of the board and is not considered an outer sub-pixel, even though it is a corner sub-pixel; it will be handled separately in the proof of Theorem 17.

The Large Tilt Graph

Before we go into the details of the algorithm, we have to tackle one issue. As was the case when deciding drainability in Section 3, computing the full tilt graph from the boundary of a board is prohibitively expensive. However, the small tilt graph developed for that purpose



■ **Figure 8** Turn gadgets and outer sub-pixels for scaling factor 3: (a) Horizontal turn gadget determined by a vertical inverse edge. (b) Vertical turn gadget determined by a horizontal inverse edge. (c) Horizontal and vertical turn gadgets at the same pixel determined by two inverse edges. (d) Potential outer sub-pixels.

is insufficient in this case, since it may contain undirected components without a sink. To find a compromise that allows computing a feasible solution using a reasonable amount of resources, we introduce the *large tilt graph* $G_L(B, S) = (V_L, E_L)$ of a board B with set of sinks S , which is another subgraph of the full tilt graph and a super-graph of G_S . In addition to the vertices of V_S , V_L contains all sinks $s \in S$ and the pixels $\{s^\ell, s^r, s^u, s^d\}$ for every sink s . Furthermore, for every reflex corner of the boundary between two segments, the leftmost and rightmost (for row segments) or topmost and bottommost (for column segments) pixels in those segments are included, as well as all pixels on intersections of row segments and column segments containing an included pixel. See Figure 4(b) for an example. G_L is the subgraph of G_F induced by V_L .

► **Lemma 11.** G_L can be constructed from the boundary of B and a set of sinks S in $\mathcal{O}(n \log n + K)$ time and $\mathcal{O}(n + K)$ space, where n is the sum of the number of corners on the boundary and the number of sinks, and K is the number of vertices arising from intersections.

Proof. TOPROVE 7 ◀

For every pixel p there is a unique pixel p^\dagger , called the *anchor* of p , defined as the pixel from G_L in the unique rectangle containing p adjacent on the inside to its upper-right corner and a single pixel from the large tilt graph adjacent on the inside to its lower-left corner. This is well-defined because two different rectangles with anchors p_1^\dagger left of p_2^\dagger would imply another pixel from G_L in the rectangle containing p_1^\dagger —either p_2^\dagger itself, if p_2^\dagger is not below p_1^\dagger , or else the pixel at the intersection of the row segment of p_1^\dagger and the column segment of p_2^\dagger . Note that if p is in G_L , then $p = p^\dagger$.

► **Lemma 12.** For every pixel p and every $x \in \{\ell, r, u, d\}$ the equality $(p^x)^\dagger = (p^\dagger)^x$ holds.

Proof. TOPROVE 8 ◀

► **Lemma 13.** For every edge (p, q) of weight w in the extended full tilt graph, there is an edge (p^\dagger, q^\dagger) of weight w in the extended large tilt graph.

Proof. TOPROVE 9 ◀

► **Observation 14.** The large tilt graph of a region is weakly connected.

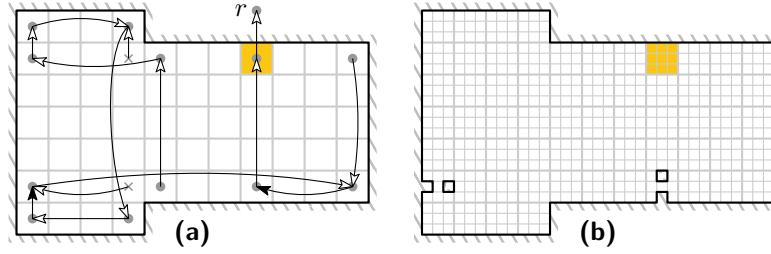
Proof. TOPROVE 10 ◀

■ **Algorithm 2** Computing a drainable sub-board of $B^{\uparrow k}$, for $k \geq 3$.

Input: The boundary of a board $B = (V, E)$ and a set $S \subseteq V$ such that every region of B contains an $s \in S$.

Output: A set of sub-pixels O such that $B^{\uparrow k}$ without O is drainable to S .

- 1 Construct G_L using Lemma 11.
- 2 Give all edges of E_L a weight of zero and add inverse edges (q, p) of weight one whenever $(p, q) \in E_L$ and $(q, p) \notin E_L$.
- 3 Add root r and edges (s, r) of weight zero for all $s \in S$.
- 4 Compute a minimum-weight arborescence T converging to r in the resulting $G_L^{\rightarrow S}$.
- 5 $O := \emptyset$
- 6 **foreach** *inverse edge* (q, p) **in** T **do**
- 7 Add the obstacles of the turn gadget determined by (q, p) to O .
- 8 **end**
- 9 **return** O



■ **Figure 9** (a) An arborescence in the extended large tilt graph of a board B with a single sink. Inverse edges are drawn with a black arrowhead. (b) The resulting obstacle placement in $B^{\uparrow 3}$.

Analysis of the Algorithm

We provide a listing of the steps in Algorithm 2 and proceed to analyze its properties and performance. Figure 9 illustrates an arborescence in the extended large tilt graph of a board B with a single sink and the resulting obstacle placement in $B^{\uparrow 3}$.

Before we prove that Algorithm 2 produces a drainable sub-board, it is important to observe that certain (inverse) edges of the extended large tilt graph cannot appear together in a minimum-weight arborescence of the large tilt graph, or only under special circumstances. Lemmas 15 and 16 not only ensure the correctness of a solution but also help bound the number of placed obstacles.

► **Lemma 15.** *For two distinct inverse edges (p_1, q_1) and (p_2, q_2) in a minimum-weight arborescence of the extended large tilt graph with their heads q_1 and q_2 in the same row segment R , at most one of their tails p_1 and p_2 can be in R .*

Proof. TOPROVE 11 ◀

► **Lemma 16.** *If there are two distinct inverse edges (p_1, q) and (p_2, q) in a minimum-weight arborescence of the extended large tilt graph, then q is a sink.*

Proof. TOPROVE 12 ◀

► **Theorem 17.** *Algorithm 2 produces a drainable sub-board of $B^{\uparrow k}$, using $\mathcal{O}((n + K) \log n)$ time and $\mathcal{O}(n + K)$ space, where n is the sum of the number of corners of the boundary and the number of sinks, and K is the number of vertices of G_L arising from intersections.*

Proof. TOPROVE 13 ◀

► **Lemma 18.** *For a board $B = (V, E)$, a set of sinks $S \subseteq V$, and a set of obstacles $O \subseteq V$ such that B without O is drainable to S , there is an arborescence in the extended full tilt graph of B of total weight at most $2|O|$.*

Proof. TOPROVE 14 ◀

Although we do all our calculations on the large tilt graph of the given board, bounding the approximation ratio requires the weight of a minimum arborescence of the extended full tilt graph of the scaled board. Lemmas 19 and 20 together show that such an arborescence cannot have smaller weight than the one we compute.

► **Lemma 19.** *If there is an arborescence of weight w in $G_F^{\leftrightarrow S}(B^{\uparrow k})$, for any $k \geq 1$, then there is an arborescence of weight at most w in $G_F^{\leftrightarrow S}(B)$.*

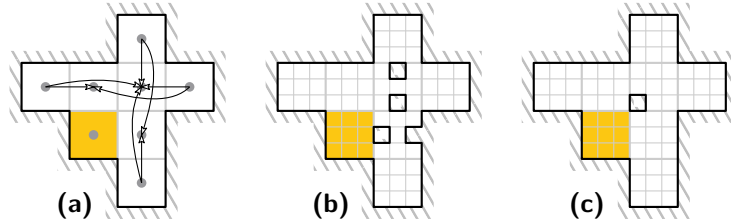
Proof. TOPROVE 15 ◀

► **Lemma 20.** *If there is an arborescence of weight w in $G_F^{\leftrightarrow S}(B)$, then there is an arborescence of weight at most w in $G_L^{\leftrightarrow S}(B, S)$.*

Proof. TOPROVE 16 ◀

► **Theorem 21.** *When applied with scaling factor 3, Algorithm 2 places at most 4 times as many obstacles as used in an optimum solution for $B^{\uparrow 3}$.*

Proof. TOPROVE 17 ◀

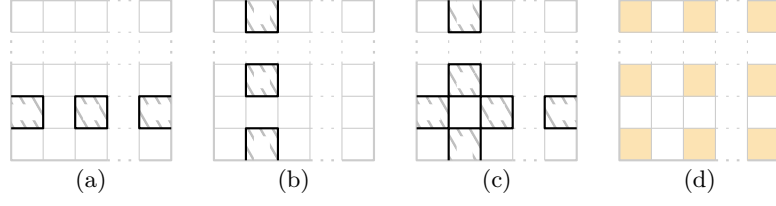


■ **Figure 10** (a) The inverse edges in $G_L^{\leftrightarrow S}(B)$ of a board B with a single sink $s \in S$. (b) Obstacles placed by Algorithm 2 for $B^{\uparrow 3}$. (c) An optimum obstacle placement for the same board and sink.

This is tight as there are 3-scaled regions for which Algorithm 2 places exactly four times as many obstacles as required, see Figure 10. For larger scaling factors we obtain the following:

► **Corollary 22.** *When applied with scaling factor $k > 3$, Algorithm 2 places at most 6 times as many obstacles as used in an optimum solution for $B^{\uparrow k}$.*

Proof. TOPROVE 18 ◀



■ **Figure 11** Turn gadgets and outer sub-pixels for scaling factor $k > 3$: Horizontal (a) and vertical (b) turn gadgets. (c) Both turn gadget at the same pixel. (d) Potential outer sub-pixels.

We leave it as an open question whether our approach can be adapted to 2-scaling. Using the same ideas, we can ensure drainability of 2-scaled regions by placing thin walls instead of obstacles, i.e., by removing edges of the underlying graph instead of vertices. It is not clear how obstacles can be placed on 2×2 super-pixels to redirect particles coming from opposite sides. An approximation may still be viable, but it will require a more sophisticated argument that will certainly lose clarity.

Furthermore, note how two perpendicular turn gadgets as placed in Figure 8(c) split a connected board into two regions that need to be drained individually. We can easily maintain connectivity by placing another obstacle at the intersection of the turn gadgets—at the cost of increasing the approximation ratio by 1. A more elaborate question is whether we can maintain the genus of the board, i.e., do not increase the number of holes, while achieving the same approximation ratio. We leave this as an open problem as well.

5 Drainability and Fillability in Generalized Models

We now step away from the full tilt model and consider arbitrary models. For a model M we define the model M^* to contain all compositions of moves from M , i.e., $m \in M^*$ if there are $m_1, m_2, \dots, m_k \in M$ such that $m = m_k \circ \dots \circ m_2 \circ m_1$, for any $k \geq 0$. We generalize the concepts of minimality and drainability in the obvious way. A move m is *monotone* if $C \subseteq D$ implies $m(C) \subseteq m(D)$; it is *volume-preserving* if $|m(C)| = |C|$ for all C . Note that the moves associated with sinks are monotone but obviously not volume-preserving. Models are called monotone or volume-preserving if all of their moves have the respective property.

The most well-studied model, apart from the full tilt model, is the *single step model*, $S1 = \{u_1, d_1, \ell_1, r_1\}$, which has particles move to an adjacent pixel in one of the four directions, unless they are blocked by the boundary or another particle. Formally, call a pixel *left-blocked* in a configuration C if it and every pixel left of it in its row segment are occupied in C . Then $p \in \ell_1(C)$ if and only if p is left-blocked in C or $p + (1, 0)^T \in C$. The other moves are defined analogously with respect to the other directions. Additionally, we introduce an extension to the full tilt and single step models that allows movement to be restricted to a subset of the segments parallel to the direction of movement. Let $M \in \{FT, S1\}$. Then the *interval extension* M_I has moves $m^{[i,j]} \in M \times \mathbb{Z}^2$ that apply the move $m \in M$ to those segments whose y -coordinate (in the case of row segments and horizontal moves) or x -coordinate (for column segments and vertical moves) lies in the interval $[i, j]$, and leave all other segments of the affected type unchanged. See the bottom half of Figure 14 for exemplary moves in FT_I . We start with an easily verified observation.

► **Observation 23.** *The models FT , $S1$, FT_I , and $S1_I$ are monotone and volume-preserving.*

► **Lemma 24.** *For every monotone model M and configurations C and D that are both reachable from V , there is a configuration $D' \subseteq D$ reachable from C .*

Proof. TOPROVE 19 ◀

► **Proposition 25.** *In a monotone model, all minimal configurations reachable from V are mutually reachable and have the same size.*

Proof. TOPROVE 20 ◀

► **Proposition 26.** *A board is drainable in a monotone model if and only if \emptyset is the only minimal configuration.*

Proof. TOPROVE 21 ◀

► **Lemma 27.** *A configuration C is minimal with respect to a set of sinks S in a volume-preserving model if and only if no $s \in S$ is occupiable from C .*

Proof. TOPROVE 22 ◀

Although the characterization of minimality in Lemma 27 applies to all volume-preserving models, its consequences vary. In the full tilt model it implies that deciding minimality is PSPACE-complete, whereas it is trivial in the single step model, as we show in Proposition 28 and Observation 30, respectively. In contrast to this, the characterization of drainability in Theorem 29 leads to a polynomial-time decision procedure for the full tilt model, as we saw when we investigated this special case in Section 3.

► **Proposition 28.** *Given a configuration C of a board B and a set S of sinks, it is PSPACE-complete to decide whether C is minimal with respect to S in the full tilt model.*

Proof. TOPROVE 23 ◀

► **Theorem 29.** *A board $B = (V, E)$ is drainable to a set of sinks S in a monotone, volume-preserving model M , if and only if, for every $p \in V$ there is $s \in S$ such that $p \rightarrow_M^* s$.*

Proof. TOPROVE 24 ◀

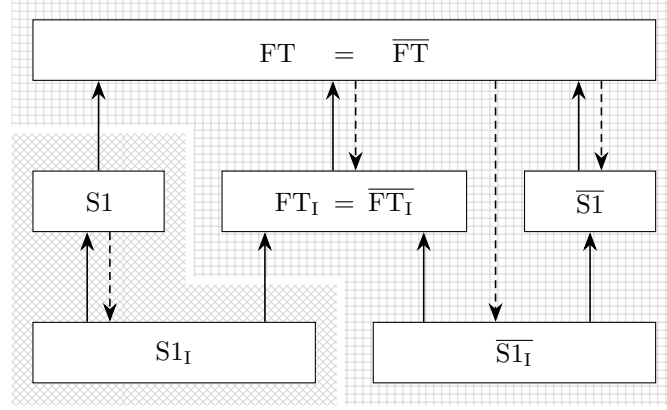
► **Observation 30.** *Every board is drainable in the single step model, as long as every one of its regions contains a sink.*

Proof. TOPROVE 25 ◀

5.1 Relative Power of Various Models

Clearly, some models are more powerful than others, in the sense that they allow us to reach more configurations. We make this notion precise by saying that a model M_2 *simulates* a model M_1 if for every $m \in M_1$ there is $m' \in M_2^*$ such that $m(C) = m'(C)$ for every configuration C . It is easy to see that S1 simulates FT (for any $m \in \{u, d, \ell, r\}$, m_∞ can be simulated using D repetitions of m_1 , where D is the maximum diameter among all regions of the board) and M_1 simulates M , for $M \in \{\text{FT}, \text{S1}\}$ (choose the interval to encompass the whole board); see the left half of Figure 12.

For the purpose of draining a board, it is useful to compare models with respect to their moves acting on single particles. To this end, we say that model M_2 *simulates* M_1 *on singletons* if, for every pixel p and move $m \in M_1$, there is $m' \in M_2^*$ such that $m(\{p\}) = m'(\{p\})$, i.e., if $p \rightarrow_{M_1} q$ implies $p \rightarrow_{M_2}^* q$, for all pixels q . Two models that simulate each other on singletons are called *equivalent on singletons*. Note that general simulation entails simulation



■ **Figure 12** Overview of the studied models and their relationships. A solid arrow from model A to model B indicates that A simulates B ; a dashed arrow indicates that A simulates B on singletons. Note that solid arrows include dashed arrows and arrows arising due to transitivity have been omitted. All models in the upper part (with the axis-parallel crosshatch pattern) are equivalent on singletons, as are the ones in the lower part (with the diagonal crosshatch pattern). Arrows on the left-hand side are derived in Section 5.1; those on the right-hand side and the equalities come from Section 5.2.

on singletons and that $S1$ and FT simulate their respective interval extensions on singletons. We now define a class of models with the property that all its members are equivalent on singletons to FT . It trivially includes FT and FT_I . Intuitively, these are the models that move at least some particles maximally, and allow doing so in all four directions.

► **Definition 31.** *A monotone and volume-preserving model M is tilt-compatible if the following conditions are satisfied for all configurations C and all occupied pixels $p \in C$.*

1. *For all $m \in M$, $\{p^\ell, p^r, p^u, p^d, p\} \cap m(C) \neq \emptyset$.*
2. *For all $x \in \{\ell, r, u, d\}$, there is $m \in M$ such that $p^x \in m(C)$.*

► **Proposition 32.** *Every tilt-compatible model is equivalent on singletons to FT .*

Proof. TOPROVE 26 ◀

► **Corollary 33.** *For every tilt-compatible model M , a board is drainable to a set of sinks S in M if and only if it is drainable to S in FT .*

5.2 Duality of Various Models

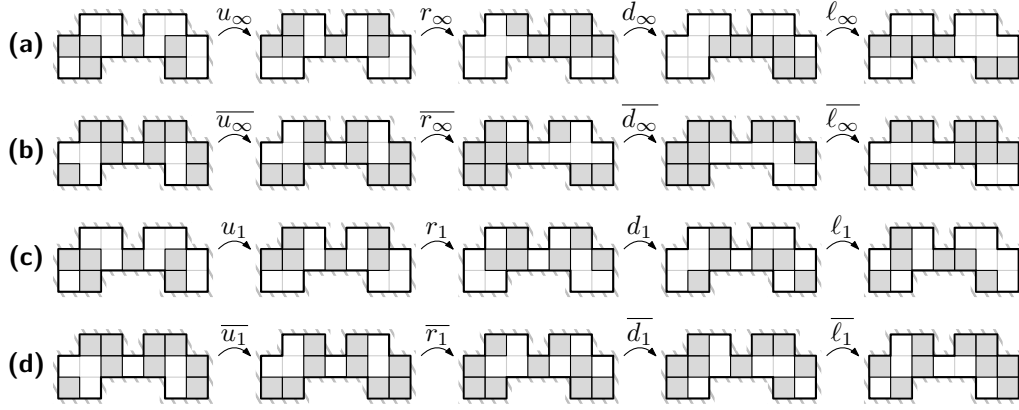
It can prove enlightening to imagine free pixels to instead be occupied by a different kind of particle, called *bubbles*, and analyze their behavior under a sequence of moves. This leads to the concept of a *dual move* for a move m , defined as $\overline{m}(C) = V \setminus m(V \setminus C)$. In the *dual model* $\overline{M} = \{\overline{m} : m \in M\}$ of a model M , particles move as bubbles do in M , and vice versa; examples are depicted in Figure 13.

► **Proposition 34.** $FT = \overline{\overline{FT}}$ and $FT_I = \overline{\overline{FT_I}}$.

Proof. TOPROVE 27 ◀

► **Lemma 35.** *For all moves m_1 and m_2 , $\overline{m_2 \circ m_1} = \overline{m_2} \circ \overline{m_1}$.*

Proof. TOPROVE 28 ◀



■ **Figure 13** Particle movement in (a) FT, (b) $\overline{\text{FT}}$, (c) S1, and (d) $\overline{\text{S1}}$. Note that every move in $\overline{\text{FT}}$ corresponds to a move in FT, whereas no such correspondence exists between $\overline{\text{S1}}$ and S1.

► **Proposition 36.** *For every model M , $\overline{M}^* = \overline{M^*}$.*

Proof. TOPROVE 29 ◀

► **Proposition 37.** *If a model M_2 simulates a model M_1 , then $\overline{M_2}$ simulates $\overline{M_1}$.*

Proof. TOPROVE 30 ◀

We now come to the reason why we introduced the concept of tilt-compatible models in the first place. It allows us to connect the full tilt model via duality to the single step model and its interval extension. An overview of the resulting relationships between the models in depicted in Figure 12.

► **Proposition 38.** *The models $\overline{\text{FT}}$, $\overline{\text{FT}}_I$, $\overline{\text{S1}}$, and $\overline{\text{S1}}_I$ are tilt-compatible.*

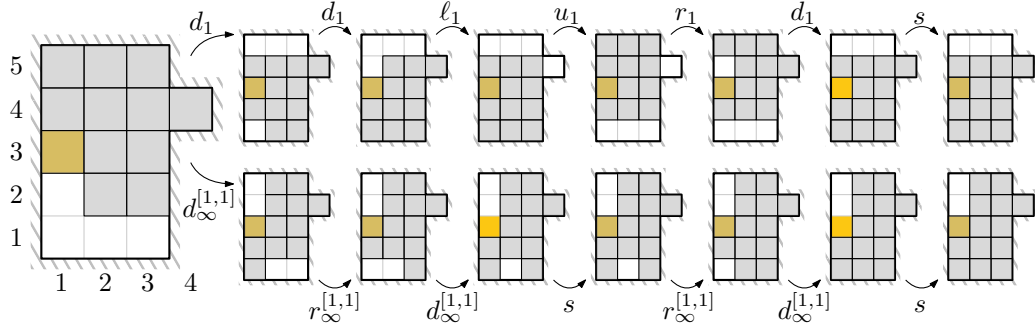
Proof. TOPROVE 31 ◀

5.3 Fillability Instead of Drainability

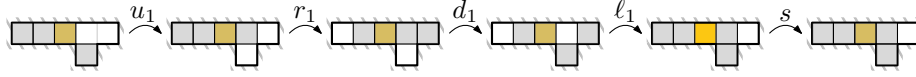
Now, instead of removing as many particles as possible from a configuration, we aim to insert as many as possible. A *source* is a move associated with a pixel $s \in V$ defined as $C \mapsto V \cup \{s\}$. Note that the dual move of a source at s is a sink at s . Analogously to the notions of minimality and drainability, a configuration C is *maximal* in a model M with respect to a set of sources S if there is no D reachable from C in $M \cup S$ with $|D| > |C|$, and a board $B = (V, E)$ is *fillable* from S if $\emptyset \rightarrow_{M \cup S}^* V$.

Due to FT_I and S1 simulating FT, it is clear that both are at least as powerful as FT when it comes to reaching large maximal configurations. Figure 14 shows that they are both strictly more powerful. In fact, S1_I is more powerful still and could, in this example, insert one more particle than FT_I . This example would seem to indicate that FT_I is more powerful than S1 but there are examples where the situation is reversed, as illustrated in Figure 15.

► **Observation 39.** *A configuration C is maximal in a model M with respect to S if and only if $V \setminus C$ is minimal in \overline{M} with respect to S . Therefore, a board is fillable from S in M if and only if it is drainable to S in \overline{M} .*



■ **Figure 14** A configuration that is maximal with respect to a source s in the full tilt model, and sequences of moves that lead to maximal configurations in S1 (top row) and FT_I (bottom row). A sequence of moves leading to the initial configuration is $m_1, r_\infty, m_1, r_\infty, m_1, m_2, \ell_\infty, m_2, \ell_\infty, m_2, u_\infty$, where m_1 is $s, d_\infty, s, d_\infty, s$, and m_2 is $u_\infty, r_\infty, d_\infty, s$.



■ **Figure 15** A configuration that is maximal with respect to a source s in FT_I, and a sequence of moves that leads to a maximal configuration in S1.

Finally, we come to the point when all the groundwork laid in the previous subsections pays off. While drainability is trivial in S1, fillability is not as obvious. Duality allows us to answer the question by considering drainability in $\bar{S1}$, which we have connected to FT.

► **Theorem 40.** *If the dual \bar{M} of a model M is tilt-compatible, then a board B is fillable from a set S in M if and only if B is drainable to S in FT.*

Proof. TOPROVE 32 ◀

Consequently, the models FT, FT_I, S1, and S1_I are all equivalent with respect to fillability. Most importantly, the results regarding drainability in the full tilt model presented in this paper, including hardness and approximation for obstacle placement, apply to fillability in all these models. We conclude by highlighting the most intriguing special case, which is quite amusing when taken out of the context of the previous subsections.

► **Corollary 41.** *A board is fillable from a set S in the single step model if and only if it is drainable to S in the full tilt model.*

6 Conclusions and Future Work

In this paper, we have analyzed ways to make a board fillable or drainable using particles moving as instructed by several models of global control signals. We have shown that placing a minimum number of obstacles to achieve fillability is NP-hard in all considered models. However, a constant-factor approximation is possible for scaled boards. The most apparent open question concerns the complexity of the obstacle placement problem for scaled boards. Is it still NP-hard or can we do better than the approximation and solve it optimally?

The next step in our future work is to investigate how to actually fill a board, once we know it is fillable. An immediate approach would be to maintain a configuration in memory and successively move bubbles to a source until the board is full. However, this may require

an amount of memory that is not polynomial in the size of an appropriate encoding of the boundary. Can we efficiently compute (short) filling sequences? Of particular interest is the worst case analysis of the length of filling sequences.

It would be interesting to better understand the geometric properties of fillable regions. In the related problem of assembly there is a hierarchy of constructable shapes [8]. These shapes derive from the *external* movement of particles, whereas ours derive from *internal* movement. Are these classifications related?

Our goal so far was to completely fill a region. Applications may impose restricted areas, i.e., positions that may never be occupied by particles, while requiring other areas to be filled. How can this constraint be handled?

Although we introduced duality of models as a tool to connect fillability to drainability, it leads to interesting new questions. Maximality in a model is strongly related to the occupancy problem in the dual model. However, occupiability in the dual model has a natural interpretation in the original model, leading to what we dub the *vacancy problem*: Given a configuration C and a pixel $p \in C$, is there a configuration D reachable from C that does not contain p ? This problem is PSPACE-complete in the full tilt model because so is the occupancy problem and FT is dual to itself. To the best of our knowledge, this natural problem has not been examined in the single step model.

► **Conjecture 42.** *The vacancy problem is PSPACE-complete in the single step model.*

A promising approach to prove the conjecture is to follow the ideas Caballero et al. [22] used to prove the relocation problem hard for the single step model. They built on the techniques of [8] and used an observation that can be seen as a precursor to our notion of duality, implying that the same methods are likely applicable in this case.

References

- 1 Ahmed Abdelkader, Aditya Acharya, and Philip Dasler. 2048 without new tiles is still hard. In *International Conference on Fun with Algorithms (FUN)*, pages 1:1–1:14, 2016. doi:10.4230/LIPIcs.FUN.2016.1.
- 2 Harold Abelson, Don Allen, Daniel Coore, Chris Hanson, George Homsy, Thomas F. Knight Jr., Radhika Nagpal, Erik Rauch, Gerald J. Sussman, and Ron Weiss. Amorphous computing. *Communications of the ACM*, 43(5):74–82, 2000. doi:10.1145/332833.332842.
- 3 Hugo A. Akitaya, Greg Aloupis, Maarten Löffler, and Anika Rounds. Trash compaction. In *European Workshop on Computational Geometry (EuroCG)*, pages 107–110, 2016. URL: https://www.eurocg2016.usi.ch/sites/default/files/paper_77.pdf.
- 4 Hugo A. Akitaya, Erik D. Demaine, Jason S. Ku, Jayson Lynch, Mike Paterson, and Csaba D. Tóth. 2048 without merging. In *Canadian Conference on Computational Geometry (CCCG)*, 2021. URL: <https://www.cccg.ca/proceedings/2021/CCCG2021.pdf>.
- 5 Hugo A. Akitaya, Maarten Löffler, and Giovanni Viglietta. Pushing blocks by sweeping lines. *Computational Geometry Topology*, 2(1):6:1–6:28, 2023. doi:10.57717/cgt.v2i1.31.
- 6 Greg Aloupis, Jean Cardinal, Sébastien Collette, Ferran Hurtado, Stefan Langerman, and Joseph O’Rourke. Draining a polygon—or rolling a ball out of a polygon. *Computational Geometry*, 47(2):316–328, 2014. doi:10.1016/J.COMGEO.2009.08.002.
- 7 Ivan J. Balaban. An optimal algorithm for finding segments intersections. In *Symposium on Computational Geometry (SoCG)*, pages 211–219, 1995. doi:10.1145/220279.220302.
- 8 Jose Balanza-Martinez, Timothy Gomez, David Caballero, Austin Luchsinger, Angel A. Cantu, Rene Reyes, Mauricio Flores, Robert T. Schweller, and Tim Wylie. Hierarchical shape construction and complexity for slidable polyominoes under uniform external forces. In *Symposium on Discrete Algorithms (SODA)*, pages 2625–2641, 2020. doi:10.1137/1.9781611975994.160.
- 9 Jose Balanza-Martinez, Austin Luchsinger, David Caballero, Rene Reyes, Angel A. Cantu, Robert Schweller, Luis Angel Garcia, and Tim Wylie. Full tilt: Universal constructors for general shapes with uniform external forces. In *Symposium on Discrete Algorithms (SODA)*, pages 2689–2708, 2019. doi:10.1137/1.9781611975482.167.
- 10 Aaron T. Becker, Erik D. Demaine, Sándor P. Fekete, Golnaz Habibi, and James McLurkin. Reconfiguring massive particle swarms with limited, global control. In *Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics (ALGOSENSORS)*, pages 51–66, 2013. doi:10.1007/978-3-642-45346-5_5.
- 11 Aaron T. Becker, Erik D. Demaine, Sándor P. Fekete, Jarrett Lonsford, and Rose Morris-Wright. Particle computation: complexity, algorithms, and logic. *Natural Computing*, 18(1):181–201, 2019. doi:10.1007/S11047-017-9666-6.
- 12 Aaron T. Becker, Erik D. Demaine, Sándor P. Fekete, and James McLurkin. Particle computation: Designing worlds to control robot swarms with only global signals. In *International Conference on Robotics and Automation (ICRA)*, pages 6751–6756, 2014. doi:10.1109/ICRA.2014.6907856.
- 13 Aaron T. Becker, Sándor P. Fekete, Phillip Keldenich, Dominik Krupke, Christian Rieck, Christian Scheffer, and Arne Schmidt. Tilt assembly: Algorithms for micro-factories that build objects with uniform external forces. *Algorithmica*, 82(2):165–187, 2020. doi:10.1007/S00453-018-0483-9.
- 14 Aaron T. Becker, Sándor P. Fekete, Li Huang, Phillip Keldenich, Linda Kleist, Dominik Krupke, Christian Rieck, and Arne Schmidt. Targeted drug delivery: Algorithmic methods for collecting a swarm of particles with uniform, external forces. In *International Conference on Robotics and Automation (ICRA)*, pages 2508–2514, 2020. doi:10.1109/ICRA40945.2020.9196551.
- 15 Aaron T. Becker, Golnaz Habibi, Justin Werfel, Michael Rubenstein, and James McLurkin. Massive uniform manipulation: Controlling large populations of simple robots with a common input signal. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 520–527, 2013. doi:10.1109/IROS.2013.6696401.

- 16 Aaron T. Becker, Yan Ou, Paul Kim, Min Jun Kim, and Agung Julius. Feedback control of many magnetized: *Tetrahymena pyriformis* cells by exploiting phase inhomogeneity. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 3317–3323, 2013. doi:10.1109/IROS.2013.6696828.
- 17 Prosenjit Bose, Efi Fogel, Tzvika Geft, Dan Halperin, and Shahar Shamaï. Optimal algorithms for separating a polyhedron from its single-part mold. *Computing in Geometry and Topology*, 3(1):7:1–7:19, 2024. doi:10.57717/cgt.v3i1.15.
- 18 Prosenjit Bose, Dan Halperin, and Shahar Shamaï. On the separation of a polyhedron from its single-part mold. In *Conference on Automation Science and Engineering (CASE)*, pages 61–66, 2017. doi:10.1109/COASE.2017.8256076.
- 19 Prosenjit Bose and Godfried T. Toussaint. Geometric and computational aspects of gravity casting. *Computer Aided Design*, 27(6):455–464, 1995. doi:10.1016/0010-4485(95)00018-M.
- 20 Prosenjit Bose, Marc J. van Kreveld, and Godfried T. Toussaint. Filling polyhedral molds. *Computer Aided Design*, 30(4):245–254, 1998. doi:10.1016/S0010-4485(97)00075-4.
- 21 David Caballero, Angel A. Cantu, Timothy Gomez, Austin Luchsinger, Robert T. Schweller, and Tim Wylie. Hardness of reconfiguring robot swarms with uniform external control in limited directions. *Journal of Information Processing*, 28:782–790, 2020. doi:10.2197/IPSJJIP.28.782.
- 22 David Caballero, Angel A. Cantu, Timothy Gomez, Austin Luchsinger, Robert T. Schweller, and Tim Wylie. Relocating units in robot swarms with uniform control signals is PSPACE-complete. In *Canadian Conference on Computational Geometry (CCCG)*, pages 49–55, 2020. URL: <https://vga.usask.ca/cccg2020/papers/Relocating%20Units%20in%20Robot%20Swarms%20with%20Uniform%20Control%20Signals.pdf>.
- 23 David Caballero, Angel A. Cantu, Timothy Gomez, Austin Luchsinger, Robert T. Schweller, and Tim Wylie. Uniform robot relocation is hard in only two directions even without obstacles. In *Unconventional Computation and Natural Computation (UCNC)*, pages 17–31, 2023. doi:10.1007/978-3-031-34034-5_2.
- 24 Erik D. Demaine and Mikhail Rudoy. A simple proof that the $(n^2 + 1)$ -puzzle is hard. *Theoretical Computer Science*, 732:80–84, 2018. doi:10.1016/J.TCS.2018.04.031.
- 25 Jakob Keller, Christian Rieck, Christian Scheffer, and Arne Schmidt. Particle-based assembly using precise global control. *Algorithmica*, 84(10):2871–2897, 2022. doi:10.1007/S00453-022-00992-2.
- 26 Matthias Konitzny, Yitong Lu, Julien Leclerc, Sándor P. Fekete, and Aaron T. Becker. Gathering physical particles with a global magnetic field using reinforcement learning. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 10126–10132, 2022. doi:10.1109/IROS47612.2022.9982256.
- 27 Neil Sarnak and Robert E. Tarjan. Planar point location using persistent search trees. *Communications of the ACM*, 29(7):669–679, 1986. doi:10.1145/6138.6151.
- 28 Robert E. Tarjan. Finding optimum branchings. *Networks*, 7(1):25–35, 1977. doi:10.1002/net.3230070103.
- 29 Craig A. Tovey. A simplified NP-complete satisfiability problem. *Discrete Applied Mathematics*, 8(1):85–89, 1984. doi:10.1016/0166-218X(84)90081-7.
- 30 Yusuke Yasui, Sara McMains, and Thomas Glau. Pool segmentation for predicting water traps. *Journal of Manufacturing Systems*, 37:494–504, 2015. doi:10.1016/j.jmsy.2014.07.006.
- 31 Yinan Zhang, Xiaolei Chen, Hang Qi, and Devin J. Balkcom. Rearranging agents in a small space using global controls. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 3576–3582, 2017. doi:10.1109/IROS.2017.8206202.