Tilt Automata: Gathering Particles with Uniform External Control

Sándor P. Fekete
Department of Computer Science, TU Braunschweig, Germany
L3S Research Center, Hannover, Germany

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

Peter Kramer
Department of Computer Science, TU Braunschweig, Germany

Jan-Marc Reinhardt
Department of Electrical Engineering and Computer Science,
Bochum University of Applied Sciences, Germany

Christian Rieck
Institute of Mathematics, University of Kassel, Germany

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

Abstract

Motivated by targeted drug delivery, we investigate the gathering of particles in the full tilt model of externally controlled motion planning: A set of particles is located at the tiles of a polyomino with all particles reacting uniformly to an external force by moving as far as possible in one of the four axis-parallel directions until they hit the boundary. The goal is to choose a sequence of directions that moves all particles to a common position. Our results include a polynomial-time algorithm for gathering in a completely filled polyomino as well as hardness reductions for approximating shortest gathering sequences and for determining whether the particles in a partially filled polyomino can be gathered. We pay special attention to the impact of restricted geometry, particularly polyominoes without holes. As a corollary, we make progress on an open question from [Balanza-Martinez et al., SODA 2020] by showing that deciding whether a given position can be occupied remains NP-hard in polyominoes without holes. Our results build on a connection we establish between tilt models and the theory of synchronizing automata.

2012 ACM Subject Classification Theory of computation → Computational geometry

Keywords and phrases Uniform control, gathering, full tilt, polyominoes, synchronizing automata

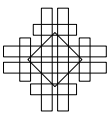
Digital Object Identifier 10.4230/LIPIcs.SoCG.2026.44

Related Version *Full Version*: `arXiv:2603.02796` [32]

Funding *Sándor P. Fekete, Jonas Friemel, Peter Kramer, Christian Scheffer*: Supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) – project SpaceAnts, 530918134. *Christian Rieck*: Supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) – 522790373.

1 Introduction

We investigate a problem motivated by targeted drug delivery [48]: How can initially dispersed drug-carrying particles be gathered at a common position? Problems of this type are closely related to advances in micro-scale robotics [50]. However, environmental constraints and the



severely limited capabilities of robotic agents at micro- and nano-scale rule out autonomous movement. Instead, a global external force, e.g., an electromagnetic field, can be used to move and steer simple robotic particles. The challenge is to use geometry to control a large swarm of particles through uniform signals within a non-uniform environment [12, 16].

Abstract mathematical models [9, 23], commonly called *tilt models* in reference to gravity as a potential external force, have been developed to tackle this problem from an algorithmic perspective. They model the environment as a *polyomino*, i.e., a finite, connected region of the square tiling, with particles positioned at tiles and moving on axis-parallel paths up, down, left, or right. Two major variants exist: In the *single step model* (S1), particles can be precisely controlled via unit steps to adjacent positions, whereas the *full tilt model* (FT) takes imprecision into account by only allowing particles to move maximally until blocked.

Algorithms and complexity results for a variety of applications [15, 25, 44, 51] have been discovered using tilt models. To investigate algorithmic approaches to targeted drug delivery, we consider particles located at the tiles of a polyomino and look for a *gathering sequence*, i.e., a sequence of moves that relocates all particles to the same tile. It is already known that gathering sequences of length $\mathcal{O}(n_c D^2)$ always exist in S1, where n_c is the number of convex corners of the polyomino and D its diameter, but finding sequences of minimum length is NP-hard [13]. Very little is known about gathering in FT, apart from the brief observation in [52] that gathering sequences do not exist for all polyominoes in FT. We remedy this situation with a comprehensive study of gathering in FT.

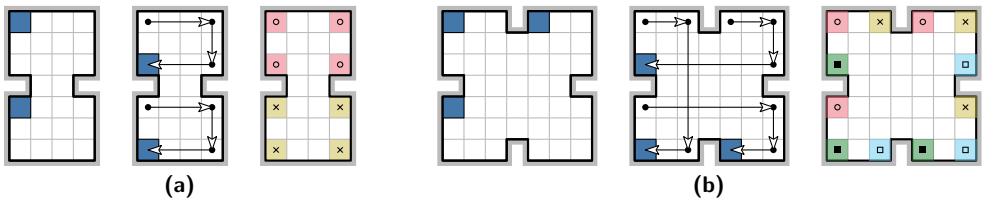


Figure 1 Two examples where gathering a set of particles is impossible in the full tilt model. Both examples depict, from left to right, a set of particles (blue squares) inside a polyomino, a sequence of moves leading to all possible configurations, and a partition of reachable positions yielding a congruence. Example **a** prohibits gathering by keeping the particles in different congruence classes, whereas the particles in **b** always occupy distinct positions of the same class.

As a first observation, we can see that the full tilt model exhibits several interesting complications that render gathering impossible. The authors of [52] already recognized that particles may become trapped in separate parts of the polyomino, see Figure 1a. However, this is not the only difficulty. Remarkably, even if all tiles reachable from the initial particle positions are mutually reachable, gathering may still be impossible, see Figure 1b.

Apart from practical applications, tilt models fascinate with simple, intuitive problems that nevertheless exhibit surprising complexity. A central question is how the computational complexity of a problem relates to the geometric complexity of the environment. Two fundamental problems are Occupancy, which asks if any of a given set of particles can be moved to a specified tile, and ShapeReconfiguration, which asks if a given configuration of particles can be transformed into another given configuration. Both are proven PSPACE-complete in FT for general polyominoes in [7], where the authors pose the complexity for *simple* polyominoes, i.e., polyominoes without holes, as an open problem. We also consider a supplementary class of polyominoes we call *mazes*: collections of horizontal and vertical lines that only intersect in T-shaped and +-shaped junctions, see Figure 2b.

1.1 Our contributions

We provide a wealth of insights into the gathering of particles in the full tilt model. Let a given polyomino have n corners, $n_c \leq n$ of which are convex. We prove the following results.

- When particles are initially positioned at every tile of a polyomino, a gathering sequence of length $\mathcal{O}(n_c n^2)$ can be computed, if one exists at all (Theorem 4 and Corollary 6).
- There are polyominoes with shortest gathering sequences of length $\Omega(n^2)$ (Theorem 7).
- Finding a shortest gathering sequence is NP-hard, even in simple mazes (Theorem 14).
- There is a polynomial-time 4-approximation for minimizing the length of a gathering sequence in simple mazes (Theorem 16), but there is no polynomial-time constant-factor approximation in general simple polyominoes (unless P = NP; Theorem 17).
- Determining whether a subset of tiles has a gathering sequence is PSPACE-complete in mazes (Corollary 13) and NP-hard in simple polyominoes (Theorem 18).

The last of these results allows us to derive progress on the open question from [7] regarding the complexity of Occupancy and ShapeReconfiguration in simple polyominoes. We show that both remain NP-hard (Corollary 19). Throughout this paper, we utilize a so far overlooked connection between tilt models and automata theory, in particular synchronizing automata. Since this topic is unlikely to be widely known among the computational geometry community, we give a brief survey in Section 3.1. We expect that more insights for tilt models can be gained by exploring this approach further. Conversely, the geometric nature of tilt models could lead to new ways to tackle open questions in automata theory. As we shall see, the discrepancy between the upper and lower bounds on the worst-case lengths of gathering sequences is closely related to a longstanding open problem known as Černý's conjecture.

Full proofs and details for statements marked by ($\star$) as well as several corollaries that exceed the space limit of this extended abstract, including results on the parameterized complexity of tilt problems, can be found in the full version [32].

1.2 Related work

Apart from the aforementioned work [13, 52] on gathering in the single step model, tilt models have been investigated for a variety of purposes. These include realizing permutations of labeled rectangular configurations [9, 10, 11, 73], simulating dual-rail logic circuits [10, 11, 64], sorting and classifying polyominoes [44], mapping an unknown region [51], and filling and draining a polyomino [33]. Notably, gathering all particles in a completely filled polyomino is strictly more difficult than draining to a freely chosen sink: While gathering requires a single position p to be reachable from everywhere, which suffices to guarantee drainability to a sink placed at p [33], this does not ensure gatherability, see Figure 1b. A major line of research is concerned with the assembly of structures from particles that can be glued together [7, 8, 15, 20, 46, 53, 63], which is strongly related to universal reconfiguration [25].

The study of the computational complexity of fundamental tilt problems was initiated in [9], where the problems now known as Occupancy and ShapeReconfiguration were originally shown NP-hard in FT. They were later settled to be PSPACE-complete [7]. In the single step model, Occupancy is solvable in polynomial time [23], whereas ShapeReconfiguration remains PSPACE-complete [24]. Restricted variants consider moves limited to two or three directions [23, 27] or deterministic move sequences [6].

Taking the maximal movement of the full tilt model and adding individual control leads to the rules of the game Ricochet Robots [30, 36]. Holzer and Schwoon [39] used related ideas in their investigation of Atomix; we cannot reuse their construction because we need a tighter analysis of the length of move sequences. Other related models examine uniform movement in a rectangle, e.g., by sweeping lines [2, 5], or the rules of the game 2048 [1, 3].

The gathering of point particles inside a polygon has been considered by Bose and Shermer [22], but their setting uses repulsion from a point within the polygon, whereas tilt models could be interpreted to use repulsion from a point at infinity. More closely related is the problem of localizing robots with severely limited capabilities [54, 57]. A sequence of movement commands that leads a robot from an initially unknown position in a polygonal environment to a definite position is equivalent to gathering all point particles in the environment at the same final position.

2 Preliminaries

In this section, we give the rigorous definitions omitted from Section 1 in favor of intuition. Every polyomino P has an associated *dual graph* $G_P = (V, E)$, see Figure 2b, where $V \subset \mathbb{Z}^2$ and $E = \{\{u, v\} : u \in V, v \in V, \|u - v\|_1 = 1\}$, and a *boundary* ∂P, an axis-aligned polygon with integer side lengths that separates V from $\mathbb{Z}^2 \setminus V$. We identify the vertices of V with the tiles of P and call them *pixels*. As a general convention throughout this text, n will denote the number of corners of ∂P, $n_c \leq n$ the number of convex corners, and $N = |V|$ the number of pixels of a given polyomino. Pixels are uniquely determined as the intersection of two *segments*, a *row segment* and a *column segment*, which are maximally contiguous parts of rows and columns; a *corner pixel* is adjacent to two perpendicular sides of ∂P, i.e., next to a convex corner, see Figure 2a. Formally, a maze is a *thin* polyomino (it contains no 2×2 squares) in which all corner pixels have degree 1, see Figure 2b.

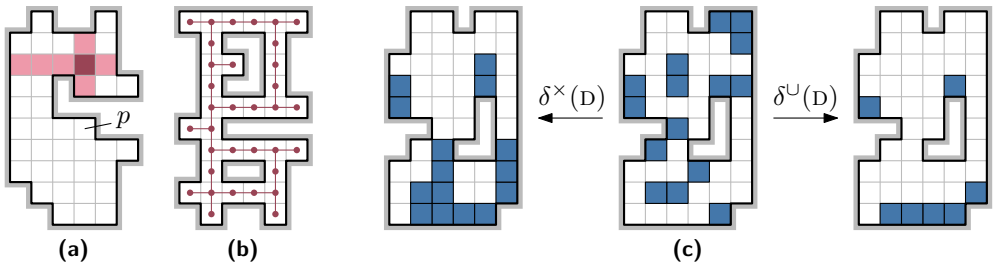


Figure 2 a A simple polyomino with a darkly shaded pixel at the intersection of two lightly shaded segments and a corner pixel p. **b** A non-simple maze and its dual graph. **c** Downwards moves in the blocking and merging variants of the full tilt model acting on a configuration.

A set $C \subseteq V$ of all pixels containing particles is called a *configuration*. Pixels $p \in C$ are *occupied*, those in $V \setminus C$ are *free*. We use the set $\mathbb{D} = \{\text{U}, \text{D}, \text{L}, \text{R}\}$ as shorthand for the directions up, down, left, and right. Formally, a tilt model defines a function $\delta : \mathbb{D} \to (2^V \to 2^V)$, i.e., it maps directions to transformations of configurations. A transformation $\delta(v)$, $v \in \mathbb{D}$, is called a *move*. δ is inductively extended to a function on $\mathbb{D}^*$, the set of (possibly empty) sequences of directions, by setting $\delta(wv) = \delta(v) \circ \delta(w)$, for $v \in \mathbb{D}$ and $w \in \mathbb{D}^*$, and $\delta(\varepsilon) = \mathrm{id}$, where ε is the empty sequence and id the identity function. Particles in the full tilt model are usually blocked when colliding with other particles, but for the purpose of gathering, they merge instead, see Figure 2c. A formal definition of the *blocking* and *merging* variants for the direction L is as follows; definitions for the other directions are analogous.

Full tilt (blocking): $p \in \delta^{\times}(\text{L})(C)$ if and only if p is one of the $|R \cap C|$ leftmost pixels of the row segment R containing p.

Full tilt (merging): $p \in \delta^{\cup}(\text{L})(C)$ if and only if p is the leftmost pixel of the row segment R containing p and $R \cap C \neq \varnothing$.

Both variants agree on singleton configurations, and the set of singleton configurations is closed under all moves. This gives rise to the function $\delta^1 : \mathbb{D} \to (V \to V)$ defined as $\delta^1(v)(p) = q$ whenever $\delta^\times(v)(\{p\}) = \delta^\cup(v)(\{p\}) = \{q\}$ for $v \in \mathbb{D}$. When δ is clear from the context, we simplify the notation by writing $C \cdot w$ instead of $\delta(w)(C)$ and $C \xrightarrow{w} C'$ to mean $C' = \delta(w)(C)$, and use moves $\delta(v)$ and directions $v \in \mathbb{D}$ interchangeably.

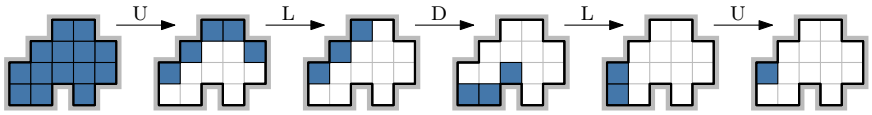


Figure 3 The sequence ULDLU is a gathering sequence for this polyomino in the full tilt model.

A *gathering sequence for a configuration* C is a sequence of directions $w \in \mathbb{D}^*$ such that $|\delta^\cup(w)(C)| = 1$, in which case C is a *gatherable configuration*; a *gathering sequence for a polyomino* P is a gathering sequence for the configuration V, see Figure 3, making P a *gatherable polyomino*. We define $\mathrm{sgs}(C)$ to be the minimum length of a gathering sequence (i.e., the length of a *shortest gathering sequence*) for a gatherable configuration C, and assign $\mathrm{sgs}(P) = \mathrm{sgs}(V)$ for a gatherable polyomino P.

The uniform movement of particles connects the merging variant to the motion of single particles: One can either consider a configuration to be gathered by merging, or move a single particle initially positioned at one of several possible positions aiming to localize it.

▶ **Observation 1.** *For* $w \in \mathbb{D}^*$ *and* $C \subseteq V$, $\delta^\cup(w)(C) = \bigcup_{p \in C} \{\delta^1(w)(p)\} = \bigcup_{p \in C} \delta^\times(w)(\{p\})$.

The single step analogue of Observation 1 is implicitly contained in [13]. Several simple consequences follow: A gathering sequence for a configuration C is also a gathering sequence for every $C' \subseteq C$. If w is a gathering sequence for C, then so is ww' for every $w' \in \mathbb{D}^*$. Prepending works for V: If w is a gathering sequence for V, then so is $w'w$ for every $w' \in \mathbb{D}^*$.

We investigate three problems in the full tilt model for a given polyomino P.

- FullGathering: Find a gathering sequence for P or decide that P is not gatherable.
- ShortestGatheringSequence: Assuming P is gatherable, compute a gathering sequence for P of minimum length $\mathrm{sgs}(P)$.
- SubsetGathering: Decide if a given configuration $C \subseteq V$ is gatherable.

We will use two algebraic properties of the full tilt model. First, every move is idempotent, i.e., $\delta(v) \circ \delta(v) = \delta(v)$ for all $v \in \mathbb{D}$. Secondly, when two moves in opposite directions are applied in sequence, the first is nullified, e.g., $\delta(\mathrm{L}) \circ \delta(\mathrm{R}) = \delta(\mathrm{L})$. Thus, we can generally assume that sequences of moves alternate between perpendicular directions.

3 Gathering particles by synchronizing automata

The notation introduced in Section 2 differs slightly from established conventions and may be considered unnecessarily heavy. It does, however, expedite a point we want to make: Tilt models are transition functions of automata. A *semi-automaton*[1] is a triple (Q, Σ, δ), where Q is a finite, non-empty set of *states*, Σ is a finite, non-empty *alphabet*, and $\delta : \Sigma \to (Q \to Q)$ is a *transition function* mapping letters of Σ to transformations of Q. As usual, δ gets extended to a function over *words* $w \in \Sigma^*$. An *acceptor* is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where (Q, Σ, δ) is

[1] All (semi-)automata considered in this text are complete, deterministic, and finite.

a semi-automaton, $q_0 \in Q$ an *initial state*, and $F \subseteq Q$ a set of *accepting states*. The *language accepted by an acceptor* A is the set of words $L(A) = \{w \in \Sigma^* : q_0 \cdot w \in F\}$. For simplicity, we will refer to both semi-automata and acceptors as *automata*. Now it is easy to see that a tilt model defines an automaton $(2^V, \mathbb{D}, \delta)$ for a polyomino P with dual graph $G_P = (V, E)$. We can even restate some established problems in the language of automata theory.

▶ **Example 2.** We are given a polyomino P, configurations $C, C' \subseteq V$, and a pixel $p \in V$.
ShapeReconfiguration: Is $L((2^V, \mathbb{D}, \delta, C, \{C'\}))$ non-empty?
Occupancy: Is $L((2^V, \mathbb{D}, \delta, C, \{C' \subseteq V : p \in C'\}))$ non-empty?

Although interesting, this connection is of no immediate use due to the exponential size of the set of states. Indeed, we would not expect a polynomial description because Occupancy and ShapeReconfiguration are PSPACE-complete in FT [7], whereas emptiness of $L(A)$ can be checked in time polynomial in the size of A [38]. The approach via automata shows its merit when we consider the analogue to gathering sequences, synchronizing words.

3.1 A brief overview of synchronizing automata

A word $w \in \Sigma^*$ is a *synchronizing word* for a set $Q' \subseteq Q$ if there is a state $q \in Q$ such that $p \xrightarrow{w} q$ for all $p \in Q'$, in which case Q' is a *synchronizing set*. The minimum length of a synchronizing word for a synchronizing set Q' is called the *reset threshold* $\mathrm{rt}(Q')$. A *synchronizing automaton* $A = (Q, \Sigma, \delta)$ has a synchronizing set of states Q, in which case we define $\mathrm{rt}(A) = \mathrm{rt}(Q)$. We now briefly summarize some important results; more information can be found in several excellent surveys [38, 60, 69, 70].

It has been (re-)discovered multiple times how to check if an automaton is synchronizing and, if so, find a synchronizing word of length $\mathcal{O}(|Q|^3)$. Eppstein's work [31] is particularly noteworthy for its careful analysis of the time and space requirements. Černý's conjecture states that $\mathrm{rt}(A) \leq (|Q| - 1)^2$ holds for every synchronizing automaton A. It is named in honor of foundational work by Černý, see [29] for a translation into English, that established the corresponding lower bound via a series of synchronizing automata. Remarkably, this seemingly simple conjecture has resisted attempts at a general solution for decades, although it has been proven for special classes of automata, e.g., Eulerian automata [43], monotonic automata [31], and others [70]. Since an improvement of the asymptotic bound $\mathcal{O}(|Q|^3)$ has proven elusive, researchers have looked at the coefficient α of the cubic term. The current best value is $\alpha \leq 0.1654$ due to Shitov [65], who expanded on ideas by Szykuła [67]. Kiefer and Ryzhikov [47] recently improved the running time of an algorithm for the more general problem of finding a word w that minimizes $|Q \cdot w|$ (called the *rank* of w).

A lower bound on the approximation ratio for $\mathrm{rt}(A)$ was improved several times [18, 19, 35], until Gawrychowski and Straszak [34] showed that $\mathrm{rt}(A)$ is NP-hard to approximate within a factor $|Q|^{1-\varepsilon}$ for every $\varepsilon > 0$. On the practical side, engineering efforts have resulted in implementations that find shortest synchronizing words for instances with over 500 states [68].

Deciding if a subset $S \subseteq Q$ is synchronizing is PSPACE-complete, as stated by Natarajan [56] and formally proven by Rystsov [59]. This problem is strongly related to IntersectionNonEmptiness: Given a family of automata over the same alphabet, decide whether they accept a common word. Kozen [49] originally proved this problem PSPACE-complete, and it remains NP-hard for tally automata [55, 66], i.e., automata over a unary alphabet.

3.2 Efficient gathering in the full tilt model

By Observation 1, a gathering sequence for a polyomino P corresponds precisely to a synchronizing word for the automaton $(V, \mathbb{D}, \delta^1)$. Thus, we could solve FullGathering by applying Eppstein's algorithms [31] to $(V, \mathbb{D}, \delta^1)$ to decide if it is synchronizing in $\mathcal{O}(N^2)$ time

and, if so, find a gathering sequence for P in $\mathcal{O}(N^3)$ time. We now sketch algorithms with running times of $\mathcal{O}(n^2)$ and $\mathcal{O}(n_c n^2)$, respectively, that use only ∂P as input. This can make a huge difference as the number of pixels N might not be polynomial in the number of corners n.

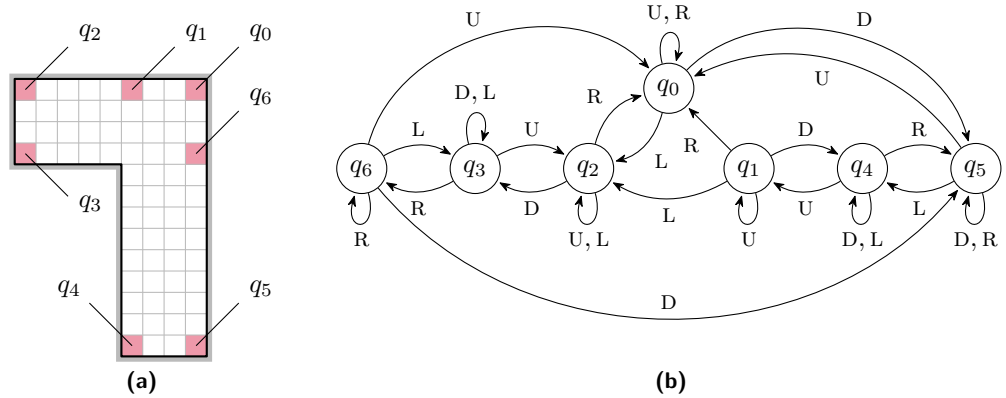


Figure 4 **a** A polyomino P and its significant pixels. **b** The full tilt automaton $A(P)$.

The key observation is that $\delta^{\cup}$ very quickly reduces the set of occupied pixels to one of size $\mathcal{O}(n)$. Consider the set $S \subseteq V$ of *significant pixels* that contains all corner pixels and possibly *helper pixels*, which are found by extending sides of ∂P that meet at reflex corners until they hit another side of ∂P. Figure 4a illustrates the significant pixels of a polyomino P with helper pixels q_1 and q_6. Observe that $|S| \leq 2n - n_c$ because at most two helper pixels are included per reflex corner. Any sequence of two moves in perpendicular directions changes the position of a particle from any pixel $p \in V$ to one in S, e.g., $\delta^1(\text{DL})(p) \in S$. S is closed under δ^1, i.e., after the move sequence DL we can focus on δ^1 restricted to S. We call the resulting automaton $A(P) = (S, \mathbb{D}, \delta^1)$ the *full tilt automaton* of P, see Figure 4b.

▶ **Lemma 3** ($\star$). *A polyomino P is gatherable if and only if the full tilt automaton $A(P)$ is synchronizing, in which case* $\text{rt}(A(P)) \leq \text{sgs}(P) \leq \text{rt}(A(P)) + 1$ *holds.*

It is a well-known fact [56, 70] that an automaton (Q, Σ, δ), with $|Q| > 1$, is synchronizing if and only if every unordered pair of states $\{s, t\} \subseteq Q$ has a synchronizing word. This suggests a strategy of iteratively merging pairs. To efficiently determine synchronizing words for pairs in $A(P)$, we construct the *pair automaton* $A(P)^{\leq 2} = (S^{\leq 2}, \mathbb{D}, \delta^{\leq 2})$, where $S^{\leq 2} = \{\{p, q\} : p, q \in S\}$ and $\delta^{\leq 2}(v)(\{p, q\}) = \{\delta^1(v)(p)\} \cup \{\delta^1(v)(q)\}$. Note that $S^{\leq 2}$ also contains all singletons $\{p\} \subseteq S$. A word w with $p \cdot w = q \cdot w$ in $A(P)$ corresponds precisely to a path labeled w from $\{p, q\}$ to $\{p \cdot w\}$ in the digraph underlying $A(P)^{\leq 2}$. Using breadth-first search on this digraph to build a shortest-path forest allows to pre-compute the next step on a shortest path to a singleton for every pair of states, if such a path exists.

To cut down the number of pairs that need to be considered, we initially reduce S to corner pixels only by repeatedly applying two perpendicular moves, e.g., DL. Observe that if $p \in S$ is a helper pixel, then $p \cdot \text{DL} < p$ in terms of their lexicographic order, as at least one coordinate decreases. Thus, $\mathcal{O}(n)$ applications of DL suffice to eliminate all helper pixels (and all corner pixels not at lower-left corners) from S. This way we need to check at most $\mathcal{O}(n_c)$ pairs, improving the running time for non-simple polyominoes with many reflex corners.

▶ **Theorem 4** ($\star$). *Given the boundary ∂P of a polyomino P with $n_c \leq n$ convex corners, a gathering sequence for P, if one exists, can be computed in $\mathcal{O}(n_c n^2)$ time.*

If we are only interested in the (non-)existence of a gathering sequence, we can stop after computing the shortest-path forest. A gathering sequence exists if and only if every pair of states has been assigned a successor on a path to a singleton [31].

▶ **Corollary 5.** *Given the boundary ∂P of a polyomino P with n corners, one can decide in $\mathcal{O}(n^2)$ time whether P is gatherable.*

3.3 Worst-case bounds for gathering sequences in FT

Once a polyomino is known to be gatherable, how long can a shortest gathering sequence be in the worst case? An upper bound matches the running time of the algorithm.

▶ **Corollary 6.** *For every gatherable polyomino P,* $\mathrm{sgs}(P) \in \mathcal{O}(n_c n^2)$.

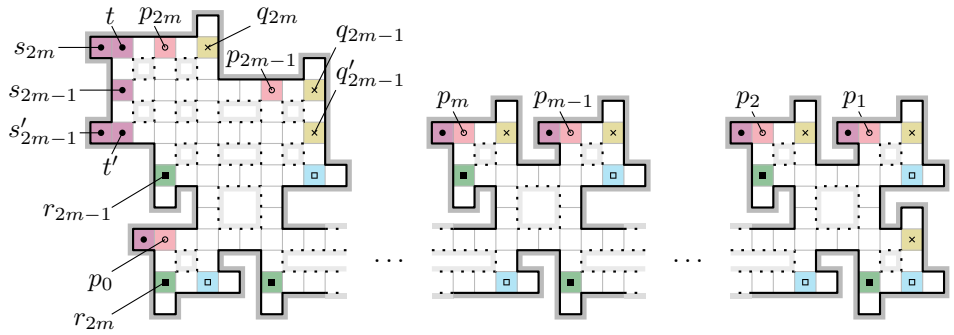


Figure 5 A family of polyominoes P_m with $\mathrm{sgs}(P_m) \in \Omega(n^2)$ in FT. Important classes of pixels are marked with distinct colors and symbols. Dotted areas indicate optional holes that can turn the simple polyomino into a maze, ensuring the construction works for both special cases.

▶ **Theorem 7** (⋆). *There is an infinite family P_m, $m \in \mathbb{N}$, of gatherable simple polyominoes such that* $\mathrm{sgs}(P_m) \in \Omega(n^2)$*, where the number of corners n grows with m.*

Proof sketch. The basic idea of the construction, depicted in Figure 5, is to place two particles on a directed cycle of length $2m + 1$, spaced apart by m positions. Moving against the direction of the cycle leads to dead ends, except at p_{2m-1}. There, LU allows to skip one step on the cycle, which can be used to reduce the distance between the particles by 1. To reduce the distance again, the whole cycle has to be traversed, resulting in $\Omega(m)$ iterations of $\Omega(m)$ moves to gather the particles, for a total of $\Omega(m^2) = \Omega(n^2)$. ◀

4 Simulating binary automata with uniformly controlled particles

If Černý's conjecture is true, then the $\Omega(n^2)$ lower bound on $\mathrm{sgs}(P)$ is actually tight.

▶ **Conjecture 8.** *For every gatherable polyomino P with n corners,* $\mathrm{sgs}(P) \in \mathcal{O}(n^2)$.

The geometric structure of polyominoes should make Conjecture 8 easier to prove than Černý's conjecture. Alas, we must dampen this hope, because particles in polyominoes can be used to simulate binary automata, i.e., automata with alphabet $\Sigma = \{0, 1\}$.

To simulate an automaton $A = (Q, \{0, 1\}, \delta)$, the main idea is to build a polyomino $P(A)$ that has every state $q \in Q$ represented by a pixel $\overline{q}$, and use fixed sequences of moves to embody transitions. A single particle is placed inside $P(A)$, its presence at $\overline{q}$ signifying that

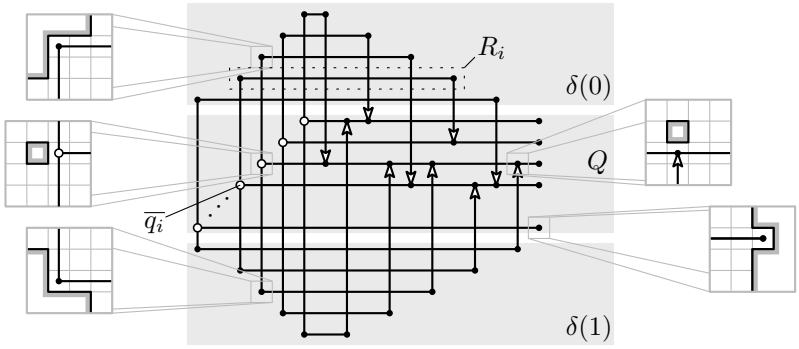


Figure 6 Schematic overview of the construction of $P(A)$ from a binary automaton A. Three blocks represent, from top to bottom, transitions for 0, states $q_i \in Q$, and transitions for 1. We zoom in to show how the schema is realized by the boundary $\partial P(A)$. See Figure 7 for a specific example.

A is in state q. We use the clockwise cycle URDL and the counter-clockwise cycle DRUL to express transitions for 0 and 1, respectively. By carefully constructing the boundary of $P(A)$, we guarantee that all minimal sequences of moves between two pixels $\overline{q_i}$ and $\overline{q_j}$ can be decomposed into combinations of URDL and DRUL, thus ensuring that minimal sequences of moves between pixels representing states are in correspondence with transitions in A.

Without loss of generality, let $Q = \{q_0, q_1, \ldots, q_{|Q|-1}\}$, and set $\overline{Q'} = \{\overline{q} : q \in Q'\}$ for sets $Q' \subseteq Q$. Our construction of $P(A)$ is schematically illustrated in Figure 6. Every state $q_i \in Q$ gets assigned a row of width $6|Q|$ including the representative pixel $\overline{q_i}$. These rows are stacked, with left-aligned rows of width $6|Q|-1$ in between, and the $\overline{q_i}$ are aligned on an upward diagonal, from $\overline{q_{|Q|-1}}$ in the bottom left up to $\overline{q_0}$. A 1×1 hole is placed to the left of every $\overline{q_i}$ except $\overline{q_{|Q|-1}}$, which is next to the outer boundary.

To implement the transitions $\delta(0)(q_i)$ in the top block, rows R_i of width $6i+3$ are stacked and aligned on the left with $\overline{q_i}$. A 1×1 hole is placed below the pixel at the intersection of the rightmost column of R_i and the row of $\overline{q_j}$ when $q_j = \delta(0)(q_i)$. This allows us to simulate a transition for 0 by the sequence of moves URDL. Similar rows and holes are constructed for $\delta(1)(q_i)$ in the bottom block, except now the rows are arranged from bottom to top and of width $6(|Q|-i)-1$, allowing us to simulate 1 by DRUL. The full construction requires $6|Q|$ convex corners and $6(3|Q|-2)$ reflex corners, for a total of $n = 12(2|Q|-1)$, and involves $N \in \mathcal{O}(|Q|^2)$ pixels. Thus, both $\partial P(A)$ and $G_{P(A)}$ can be computed in polynomial time.

For a specific example consider the binary automaton A_0 that accepts the words containing exactly one 0, see Figure 7a. The polyomino $P(A_0)$ resulting from our construction is depicted in Figure 7b, where marked pixels show congruence classes of pixels reachable from $\overline{Q}$, in particular those traversed on cycles URDL and DRUL.

▶ **Observation 9.** *For all $p, q \in Q$, $p \xrightarrow{0} q \Leftrightarrow \overline{p} \xrightarrow{\text{URDL}} \overline{q}$ and $p \xrightarrow{1} q \Leftrightarrow \overline{p} \xrightarrow{\text{DRUL}} \overline{q}$.*

▶ **Lemma 10** (⋆). *For all $p \in Q$, with representative pixels $\overline{p} \in P(A)$, and all $w \in \mathbb{D}^*$ with $\overline{p} \cdot w \in \overline{Q}$, there is $w' \in \{\text{URDL}, \text{DRUL}\}^*$ with $|w'| \leq |w|$ that satisfies $\overline{q} \cdot w' = \overline{q} \cdot w$ for all $q \in Q$.*

▶ **Theorem 11.** *A set $S \subseteq Q$ of states of an automaton $A = (Q, \{0,1\}, \delta)$ is synchronizing if and only if its set of representatives $\overline{S}$ is gatherable in $P(A)$, in which case $\operatorname{sgs}(\overline{S}) = 4\operatorname{rt}(S)$.*

Proof. For the forward direction, let $w \in \{0,1\}^*$ be a synchronizing word for S of minimum length. Use Observation 9 to build a gathering sequence $\overline{w} \in \{\text{URDL}, \text{DRUL}\}^*$ for $\overline{S}$ satisfying $|\overline{w}| = 4|w|$. Assume, for sake of contradiction, that there is a gathering sequence $\overline{w}' \in \mathbb{D}^*$

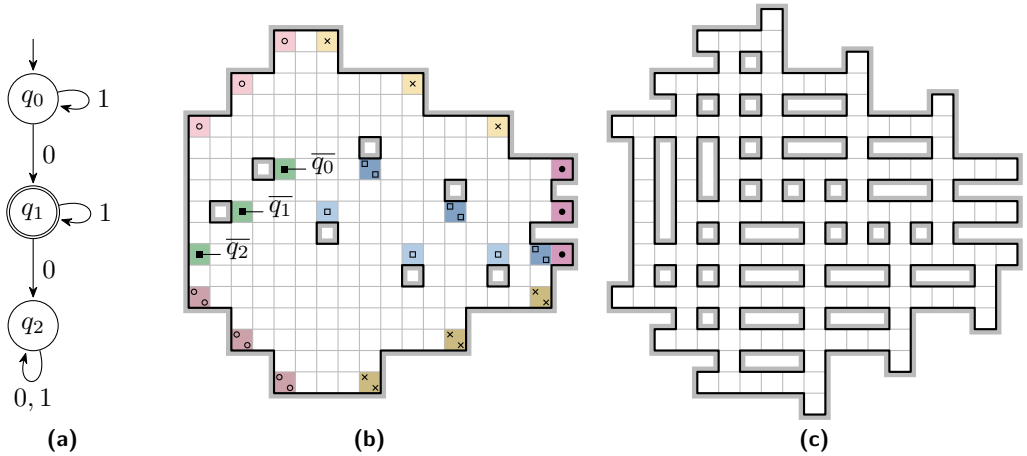


Figure 7 **a** The binary automaton A_0 that accepts all words containing exactly one 0. **b** The polyomino $P(A_0)$. Colors and symbols indicate congruence classes of pixels reachable from representatives $\overline{q_i}$. **c** $P(A_0)$ turned into a maze $M(A_0)$ by adding holes and pixels of degree 1.

with $|\overline{w}'| < |\overline{w}|$. First apply Lemma 10 to $\overline{w}'$ and then Observation 9 to get a synchronizing word $w' \in \{0,1\}^*$ for S with $|w'| < |w|$, contradicting the assumption that w has minimum length. The backward direction is very similar. ◀

▶ **Corollary 12** (⋆). *If* $\mathrm{sgs}(P) \in \mathcal{O}(n^2)$ *for every gatherable polyomino* P *with* n *corners, then every synchronizing binary automaton* $A = (Q, \{0,1\}, \delta)$ *has* $\mathrm{rt}(A) \in \mathcal{O}(|Q|^2)$.

While a quadratic upper bound on the reset threshold for synchronizing binary automata would not settle Černý's conjecture, it would be remarkable progress. After decades of research, the existence of such an upper bound is still an open problem, suggesting that a proof for Conjecture 8 may be hard to find.

It is easy to adapt the construction of $P(A)$ to produce a maze $M(A)$, at the cost of possibly requiring $\Omega(|Q|^2)$ corners, by adding more holes as well as pixels of degree 1 next to corner pixels, as illustrated for $M(A_0)$ in Figure 7c.

▶ **Corollary 13.** SubsetGathering *is* PSPACE*-complete, even in mazes.*

Proof. It is easy to see that a non-deterministic algorithm for SubsetGathering could run in polynomial space, which implies containment in NPSPACE = PSPACE [62].

To prove PSPACE-hardness, we reduce from SubsetSynchronizability: Given an automaton $A = (Q, \Sigma, \delta)$ and a set $S \subseteq Q$, determine if S is synchronizing. This problem was originally shown PSPACE-hard by Rystsov [59], and Vorel [72] proved it remains so for $\Sigma = \{0,1\}$. We construct $P(A)$ for a given binary automaton $A = (Q, \{0,1\}, \delta)$ in polynomial time. By Theorem 11, $S \subseteq Q$ is synchronizing if and only if $\overline{S}$ is gatherable. This property remains true when $P(A)$ is turned into a maze $M(A)$. ◀

We want to note that PSPACE-hardness of SubsetGathering in general polyominoes also follows from [7], but we welcome an alternative proof in keeping with our automata theme.

5 Approximating gathering sequences in simple polyominoes

We first show that ShortestGatheringSequence is NP-hard even in simple mazes. To do this, we reduce from ShortestCommonSupersequence where, given an alphabet Σ and a set S of words over Σ, we want to find a shortest word $\overline{w}$ such that for each $w \in S$, there

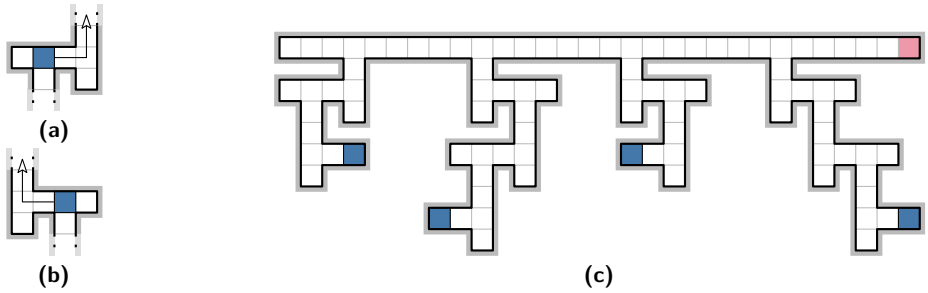


Figure 8 **a** 0-gadget, move sequence RU. **b** 1-gadget, move sequence LU. **c** The gathering instance for the words 10, 001, 01, and 111. A shortest supersequence is 10011 and the corresponding shortest gathering sequence is LU RU RU LU LU R. The particles gather in the red pixel.

exists a sequence $i_1 < \cdots < i_{|w|}$ with $w_j = \overline{w}_{i_j}$ for each j, i.e., $\overline{w}$ is a supersequence of each word $w \in S$. This problem is already NP-hard if $\Sigma = \{0, 1\}$ [58]. Gerbush and Heeringa [35] used a reduction from a generalized problem to prove inapproximability of the reset threshold of synchronizing automata, which inspired our approach.

▶ **Theorem 14.** ShortestGatheringSequence *is* NP*-hard in simple mazes.*

Proof. Given a set S of words over $\{0, 1\}$, $|S| > 1$, we construct a simple maze as follows: We start with a (sufficiently long) row segment R. For each word $w \in S$, we create a vertical path gadget consisting of the symbol gadgets shown in Figures 8a and 8b in the order imposed by w. We attach this gadget to the bottom of R, see Figure 8c. Clearly, this construction is a maze without holes. In each path gadget, we focus on the particle furthest from R as any gathering sequence for these particles is also a gathering sequence for the full polyomino.

We show that there exists a supersequence for S of length k if and only if there exists a gathering sequence of length $2k + 1$ in the corresponding maze. Let $\overline{w} \in \{0, 1\}^*$ be a supersequence for S with $|\overline{w}| = k$. We replace each 0 in $\overline{w}$ with the moves RU and each 1 with LU. In each of the path gadgets, the particle progresses to the next symbol gadget if and only if the symbol in $\overline{w}$ aligns with the moves for its current symbol gadget. Otherwise, it remains in its row. Thus, after the $2k$ moves, all particles have progressed to the row segment R. To gather them in R's rightmost pixel, we add one final R move.

Conversely, let $\overline{w}' \in \mathbb{D}^*$ be a gathering sequence. Without loss of generality, let $|\overline{w}'|$ be minimal. Then, $\overline{w}'$ alternates between horizontal and vertical moves and the first move is horizontal because a vertical move has no effect in the first step. Additionally, $\overline{w}'$ does not contain any D-move as particles can only gather in the row segment R and all shortest sequences to any pixel in R consist of only the moves U, L, and R. Finally, since $|S| > 1$, the last move is horizontal as particles from different path gadgets have to gather in the leftmost or rightmost pixel of R. Therefore, $\overline{w}'$ is a sequence of length $2k + 1$ for some k, consisting of RU and LU subsequences and ending with a horizontal move. We ignore the final move and replace each RU with a 0 and each LU with a 1 to construct a word $\overline{w}$ of length k. As before, a particle progresses to the next symbol gadget exactly if the word in the path gadget contains the corresponding symbol at the position of the particle's gadget. Therefore, since $\overline{w}'$ is a gathering sequence and all particles end up in R, $\overline{w}$ is a supersequence for S. ◀

On the other hand, we can approximate the length of a shortest gathering sequence to within a factor of 4 in simple mazes. To achieve this, we first observe that all such instances have exactly one row or column segment where the particles can gather. Throughout the remainder of this section, we only consider segments consisting of at least two pixels.

▶ **Lemma 15** (⋆). *Let P be a gatherable simple maze with $N > 1$ pixels. Then all row or column segments of P have at least one end point that is a corner pixel and exactly one segment has two corner pixel end points.*

▶ **Theorem 16.** *Let P be a gatherable simple maze. Then we can compute a gathering sequence for P of length at most $4 \cdot \mathrm{sgs}(P)$ in polynomial time.*

Proof. Assume $N > 1$. By Lemma 15, there exists a unique segment R whose end points are both corner pixels and any other segment has exactly one corner pixel. Clearly, all particles have to gather in R as particles initially located in R's corner pixels cannot enter any other segment. It suffices to focus on particles located at corner pixels as all other particles are also gathered by gathering sequences for corner pixels in simple mazes. Now consider a particle p located in the corner pixel of a horizontal segment $S_1 \neq R$. To reach R, p has to pass a sequence of alternating horizontal and vertical segments $S_1, \ldots, S_\ell$ with $S_\ell = R$. Therefore, the shortest sequence moving p to R consists of alternating $\{\mathrm{L}, \mathrm{R}\}$ and $\{\mathrm{U}, \mathrm{D}\}$ moves. Note that p cannot return to S_i after progressing to S_{i+1} and then moving in any direction perpendicular to S_i because the intersection pixel between any consecutive segments S_i, S_{i+1} has degree 3 by the definition of mazes.

Let d be the maximum length of a shortest sequence from any corner pixel to R and consider the sequence URDL. Any particle initially located at a corner pixel progresses to the next segment with at least one of the four moves. Thus, the sequence $(\mathrm{URDL})^d$ is at most four times as long as the shortest sequence moving all particles to R. Finally, the last move in any gathering sequence is perpendicular to R to gather all particles in one of R's end points. ◀

For arbitrary simple polyominoes, there is likely no polynomial-time algorithm approximating $\mathrm{sgs}(P)$ within any constant factor. To show this, we extend our reduction from ShortestCommonSupersequence to unbounded alphabets. If the alphabet size is not fixed, the problem is hard to approximate within any constant factor $\alpha \geq 1$ unless P = NP [41].

We can use the same approach to construct a polyomino as in Theorem 14, i.e., we attach a path gadget consisting of symbol gadgets for each given word to a baseline segment where particles gather. It suffices to generalize the symbol gadget for larger alphabets.

▶ **Theorem 17** (⋆). *Unless* P = NP*, there is no polynomial-time approximation algorithm for* ShortestGatheringSequence *in simple polyominoes with approximation ratio $\alpha \in \mathcal{O}(1)$.*

6 Gathering subsets in simple polyominoes

Like previous reductions for the full tilt model [7, 9], our PSPACE-hardness proof for SubsetGathering in Corollary 13 requires polyominoes with holes. We can retain a weaker hardness result for simple polyominoes by reducing from IntersectionNonEmptiness of *tally automata*, i.e., automata over the unary alphabet $\{0\}$: Given a family $A_i = (Q_i, \{0\}, \delta_i, q_i, F_i)$ of tally automata, determine whether $\bigcap_i L(A_i) \neq \varnothing$, i.e., whether the A_i accept a common word. The NP-hardness of this variant goes back to seminal work by Stockmeyer and Meyer [66]. A newer, independent proof by Morawietz, Rehs, and Weller [55] emphasizes that this variant remains NP-hard if the graphs underlying the automata are cycles. Inspection of the proof reveals that we may also assume the lengths of all cycles to be odd and greater than 1, since they are chosen to be products of large primes. Finally, we may assume all automata to have at least one non-accepting state (or the automaton would accept all words). We implicitly include these restrictions when talking about tally automata in the following.

Given a family $A_i = (Q_i, \{0\}, \delta_i, q_i, F_i)$, $1 \leq i \leq k$, of k tally automata, we construct a simple polyomino $P(A_1, \ldots, A_k)$ and an initial configuration as follows.

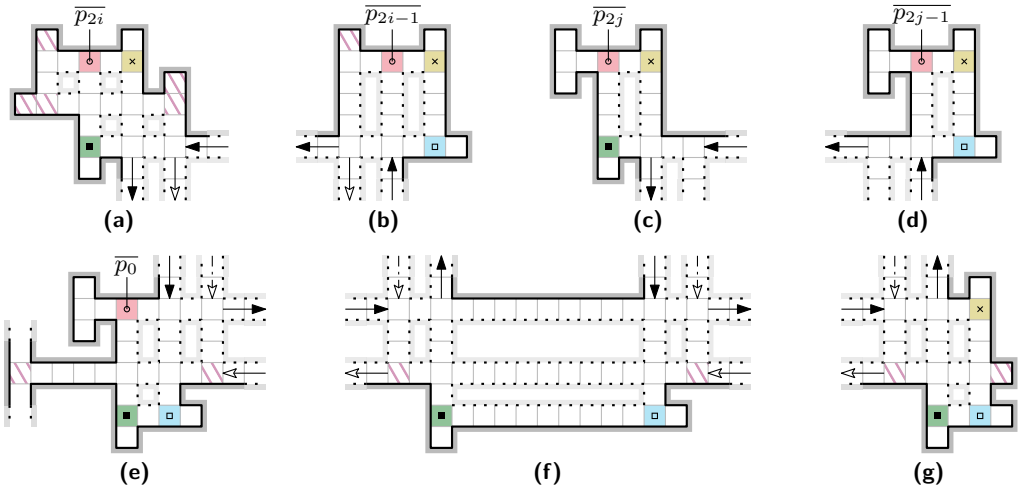


Figure 9 Gadgets used in the reduction from tally automata intersection. Pixels marked with colors and symbols indicate the cycle RDLU between pixels representing states, which is continued between gadgets via the black arrows. The stroked tiling pattern highlights pixels with a safe path to the goal area using the white arrows. Dotted areas specify optional holes that turn the simple polyomino into a maze. **a** Accepting state with even index. **b** Accepting state with odd index. **c** Rejecting state with even index. **d** Rejecting state with odd index. **e** Start gadget including $\overline{p_0}$ and access to the goal area to the left. **f** Bridge gadget. **g** End gadget connecting $\overline{p_0}$ to $\overline{p_1}$.

Consider an automaton A_i with $|Q_i| = 2m + 1$ for some $m \geq 1$. Number its states $p_0, \ldots, p_{2m}$ such that $p_0 \notin F_i$ and $p_j \to p_{(j+1) \bmod |Q_i|}$ for all $0 \leq j < |Q_i|$. Place a *start gadget* containing a pixel $\overline{p_0}$ representing p_0, followed horizontally by $m-1$ *bridge gadgets* and a final *end gadget*; see Figures 9e–9g. Above the partial construction, $2m$ gadgets are placed, from right to left, for the remaining states $p_j \in \{p_1, \ldots, p_{2m}\}$, depending on two factors: the parity of j and whether p_j is accepting. Gadgets for the four possible combinations are depicted in Figures 9a–9d, each containing a representative pixel $\overline{p_j}$.

Partial constructions for A_i are stacked vertically and connected to a *goal area* to the left, which is simply a column segment ending in pixels of degree 1. For a set $S \subseteq \bigcup_i Q_i$, we define $\overline{S} = \{\overline{q} : q \in S\}$ to be the set of representative pixels for states in S. The initial configuration is $C_0 = \bigcup_i \{\overline{q_i}\}$, i.e., the representative pixels for the initial states. Figure 10 depicts a small but complete example of the construction for two tally automata.

▶ **Theorem 18** (⋆). SubsetGathering *is* NP*-hard in simple polyominoes.*

Proof sketch. A common word $0^\ell \in \bigcap_i L(A_i)$ leads to a gathering sequence $(\text{RDLU})^\ell$ LDRDLD for C_0 in $P(A_1, \ldots, A_k)$. For the reverse direction, we argue that if there is any gathering sequence for C_0, then it must first move all particles to representatives of accepting states, which can be done with a (shorter) sequence of the form $(\text{RDLU})^{\ell'}$, which corresponds to a common word $0^{\ell'}$ for the automata. Since the reduction uses a polynomial number of constant-size gadgets, it can be executed in polynomial time. ◀

▶ **Corollary 19.** Occupancy *and* ShapeReconfiguration *are* NP*-hard in the blocking variant of* FT*, even in simple polyominoes.*

Proof. Take the construction from Theorem 18 for k automata and extend the goal area upwards so that there are k pixels above the topmost intersection. The distinction between $\delta^\times$ and $\delta^\cup$ only matters in the goal area, since that is the only segment that can possibly

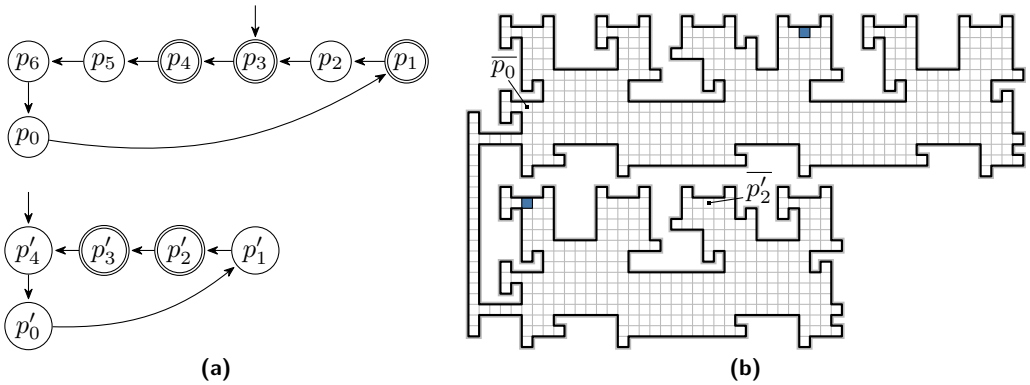


Figure 10 Instance of SubsetGathering produced by the reduction from tally automata intersection. **a** Two tally automata, A_1, A_2; both accept the word of length 8. **b** The simple polyomino $P(A_1, A_2)$; $(\text{RDLU})^8$ LDRDLD is a gathering sequence for its initial configuration C_0.

Table 1 Overview of complexity results for gathering problems in FT, unless P = NP.

Problem	Geometry	Complexity	Reference
FullGathering	General / Simple	$\mathcal{O}(n_c n^2)$ / $\Omega(n^2)$	**Thms. 4, 7**
		$\mathcal{O}(n^2)$ conj.	**Conj. 8**
ShortestGatheringSequence	Simple	No $\mathcal{O}(1)$-approx.	**Thm. 17**
	Simple maze	NP-hard, 4-approx.	**Thms. 14, 16**
SubsetGathering	Maze	PSPACE-complete	**Cor. 13**
	Simple	NP-hard	**Thm. 18**

contain more than one particle at a time. Now the kth pixel from the top in the goal area can be occupied – and the configuration consisting of the k topmost pixels in the goal are can be reached – if and only if the automata accept a common word. ◀

7 Conclusions

See Table 1 for an overview of our results. The reduction from Section 6 transfers to the related problem of *collecting* [52] a given set of particles into a *connected* configuration in the blocking variant of FT. Our hardness results also apply if the particles are to be gathered at a specified position, since one could simply check all possible positions if this was practical. In the case of FullGathering, it is still possible to find a gathering sequence $w \in \mathbb{D}^*$ such that $V \xrightarrow{w} \{p\}$ for a given pixel p in polynomial time, or decide that no such sequence exists: Assume there are two distinct pixels q_1, q_2 such that $V \xrightarrow{w_i} \{q_i\}$ for $w_1, w_2 \in \mathbb{D}^*$. Then $\{q_1\} \xrightarrow{w_2} \{q_2\}$ and $\{q_2\} \xrightarrow{w_1} \{q_1\}$, i.e., all pixels where particles can be gathered are mutually reachable. Thus, one can check if all particles can be gathered at a given pixel p by using the techniques from Section 3 to find a gathering sequence targeting an unspecified pixel q, and then check if p is reachable from q. If either step fails gathering at p is impossible.

It may be worth reiterating that our algorithmic results are polynomial in the number n of corners of a given polyomino, whereas our hardness reductions produce the dual graph with N pixels, i.e., the respective problems are hard even for a unary representation of space.

A number of open questions remain. While closing the gap between the worst-case bounds $\Omega(n^2)$ and $\mathcal{O}(n_c n^2)$ for $\text{sgs}(P)$ in general polyominoes looks very hard due to its connection to Černý's conjecture, it may be feasible to close the gap in special classes of polyominoes. Is it easier to prove an upper bound $\text{sgs}(P) \in \mathcal{O}(n^2)$ for simple polyominoes?

We have shown SubsetGathering, Occupancy, and ShapeReconfiguration to be NP-hard in simple polyominoes but did not definitively settle their complexity. Can these problems be shown to remain PSPACE-complete in simple polyominoes?

ShortestGatheringSequence can be approximated within a constant factor in simple mazes, which is NP-hard in general simple polyominoes. Is SubsetGathering also easier in simple mazes? We conjecture it to be polynomially solvable in this case.

References

1 Ahmed Abdelkader, Aditya Acharya, and Philip Dasler. 2048 without new tiles is still hard. In *Fun with Algorithms (FUN)*, pages 1:1–1:14, 2016. doi:10.4230/LIPIcs.FUN.2016.1.

2 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.

3 Hugo A. Akitaya, Erik D. Demaine, Jason S. Ku, Jayson Lynch, and Csaba D. Tóth. 2048 without merging. In *Canadian Conference on Computational Geometry (CCCG)*, pages 285–291, 2020. URL: http://vga.usask.ca/cccg2020/papers/2048%20Without%20Merging.pdf.

4 Hugo A. Akitaya, Maarten Löffler, and Giovanni Viglietta. Pushing blocks by sweeping lines. In *Fun with Algorithms (FUN)*, pages 1:1–1:21, 2022. doi:10.4230/LIPIcs.FUN.2022.1.

5 Hugo A. Akitaya, Maarten Löffler, and Giovanni Viglietta. Pushing blocks by sweeping lines. *Computation in Geometry and Topology*, 2(1):6:1–6:28, 2023. A conference version of this paper appeared at FUN 2022 [4]. doi:10.57717/cgt.v2i1.31.

6 Alberto Avila-Jimenez, David Barreda, Sarah-Laurie Evans, Austin Luchsinger, Aiden Massie, Robert Schweller, Evan Tomai, and Tim Wylie. General computation using slidable tiles with deterministic global forces. In *Innovations in Theoretical Computer Science (ITCS)*, pages 14:1–14:25, 2026. doi:10.4230/LIPIcs.ITCS.2026.14.

7 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.

8 Jose Balanza-Martinez, Austin Luchsinger, David Caballero, Rene Reyes, Angel A. Cantu, Robert T. 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.

9 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.

10 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. Conference versions related to this paper appeared at ALGOSENSORS 2013 [9], ICRA 2014 [11], and ICRA 2015 [64]. doi:10.1007/S11047-017-9666-6.

11 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.

12 Aaron T. Becker, Erik D. Demaine, Sándor P. Fekete, Hamed Mohtasham Shad, and Rose Morris-Wright. Tilt: The video – designing worlds to control robot swarms with only global signals. In *International Symposium on Computational Geometry (SoCG)*, pages 16–18, 2015. doi:10.4230/LIPIcs.SOCG.2015.16.

13 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.

14 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. In *International Symposium on Algorithms and Computation (ISAAC)*, pages 11:1–11:13, 2017. doi:10.4230/LIPIcs.ISAAC.2017.11.

15 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. A conference version related to this paper appeared at ISAAC 2017 [14]. doi:10.1007/S00453-018-0483-9.

16 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.

17 Mikhail V. Berlinkov. Approximating the minimum length of synchronizing words is hard. In *International Computer Science Symposium in Russia (CSR)*, pages 37–47, 2010. doi:10.1007/978-3-642-13182-0_4.

18 Mikhail V. Berlinkov. Approximating the minimum length of synchronizing words is hard. *Theory of Computing Systems*, 54(2):211–223, 2014. A conference version of this paper appeared at CSR 2010 [17]. doi:10.1007/S00224-013-9511-Y.

19 Mikhail V. Berlinkov. On two algorithmic problems about synchronizing automata (short paper). In *Developments in Language Theory (DLT)*, pages 61–67, 2014. doi:10.1007/978-3-319-09698-8_6.

20 Patrick Blumenberg, Arne Schmidt, and Aaron T. Becker. Computing motion plans for assembling particles with global control. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 7296–7302, 2023. doi:10.1109/IROS55552.2023.10341556.

21 Prosenjit Bose and Thomas C. Shermer. Gathering by repulsion. In *Scandinavian Symposium and Workshops on Algorithm Theory (SWAT)*, pages 13:1–13:12, 2018. doi:10.4230/LIPIcs.SWAT.2018.13.

22 Prosenjit Bose and Thomas C. Shermer. Gathering by repulsion. *Computational Geometry*, 90:101627, 2020. A conference version of this paper appeared at SWAT 2018 [21]. doi:10.1016/J.COMGEO.2020.101627.

23 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.

24 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.

25 David Caballero, Angel A. Cantu, Timothy Gomez, Austin Luchsinger, Robert T. Schweller, and Tim Wylie. Fast reconfiguration of robot swarms with uniform control signals. *Natural Computing*, 20(4):659–669, 2021. doi:10.1007/S11047-021-09864-0.

26 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.

27 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. *Natural Computing*, 24(1):3–16, 2025. A conference version of this paper appeared at UCNC 2023 [26]. doi:10.1007/S11047-024-10007-4.

28 Ján Černý. Poznámka k homogénnym experimentom s konečnými automatmi. *Matematicko-fyzikálny časopis*, 14(3):208–216, 1964. URL: https://dml.cz/dmlcz/126647.

29 Ján Černý. A note on homogeneous experiments with finite automata. *Journal of Automata, Languages and Combinatorics*, 24(2–4):123–132, 2019. The original article was published in 1964 in Slovak [28]. English translation by Markus Holzer and Bianca Truthe. doi:10.25596/jalc-2019-123.

30 Birgit Engels and Tom Kamphans. Randolphs robot game is NP-hard! *Electronic Notes in Discrete Mathematics*, 25:49–53, 2006. doi:10.1016/J.ENDM.2006.06.062.

31 David Eppstein. Reset sequences for monotonic automata. *SIAM Journal on Computing*, 19(3):500–510, 1990. doi:10.1137/0219033.

32 Sándor P. Fekete, Jonas Friemel, Peter Kramer, Jan-Marc Reinhardt, Christian Rieck, and Christian Scheffer. Tilt automata: Gathering particles with uniform external control. *CoRR*, 2026. arXiv:2603.02796.

33 Sándor P. Fekete, Peter Kramer, Jan-Marc Reinhardt, Christian Rieck, and Christian Scheffer. Drainability and fillability of polyominoes in diverse models of global control. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 74:1–74:19, 2025. doi:10.4230/LIPIcs.ICALP.2025.74.

34 Paweł Gawrychowski and Damian Straszak. Strong inapproximability of the shortest reset word. In *Mathematical Foundations of Computer Science (MFCS)*, pages 243–255, 2015. doi:10.1007/978-3-662-48057-1_19.

35 Michael Gerbush and Brent Heeringa. Approximating minimum reset sequences. In *Conference on Implementation and Application of Automata (CIAA)*, pages 154–162, 2010. doi:10.1007/978-3-642-18098-9_17.

36 Adam Hesterberg and Justin Kopinsky. The parameterized complexity of ricochet robots. *Journal of Information Processing*, 25:716–723, 2017. doi:10.2197/IPSJJIP.25.716.

37 Markus Holzer and Martin Kutrib. Descriptional and computational complexity of finite automata. In *Language and Automata Theory and Applications (LATA)*, pages 23–42, 2009. doi:10.1007/978-3-642-00982-2_3.

38 Markus Holzer and Martin Kutrib. Descriptional and computational complexity of finite automata—A survey. *Information and Computation*, 209(3):456–470, 2011. A conference version of this paper appeared at LATA 2009 [37]. doi:10.1016/J.IC.2010.11.013.

39 Markus Holzer and Stefan Schwoon. Assembling molecules in ATOMIX is hard. *Theoretical Computer Science*, 313(3):447–462, 2004. doi:10.1016/J.TCS.2002.11.002.

40 Tao Jiang and Ming Li. On the approximation of shortest common supersequences and longest common subsequences. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 191–202, 1994. doi:10.1007/3-540-58201-0_68.

41 Tao Jiang and Ming Li. On the approximation of shortest common supersequences and longest common subsequences. *SIAM Journal on Computing*, 24(5):1122–1139, 1995. A conference version of this paper appeared at ICALP 1994 [40]. doi:10.1137/S009753979223842X.

42 Jarkko Kari. Synchronizing finite automata on Eulerian digraphs. In *Mathematical Foundations of Computer Science (MFCS)*, pages 432–438, 2001. doi:10.1007/3-540-44683-4_38.

43 Jarkko Kari. Synchronizing finite automata on Eulerian digraphs. *Theoretical Computer Science*, 295:223–232, 2003. A conference version of this paper appeared at MFCS 2001 [42]. doi:10.1016/S0304-3975(02)00405-X.

44 Phillip Keldenich, Sheryl Manzoor, Li Huang, Dominik Krupke, Arne Schmidt, Sándor P. Fekete, and Aaron T. Becker. On designing 2D discrete workspaces to sort or classify polyominoes. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 1–9, 2018. doi:10.1109/IROS.2018.8594150.

45 Jakob Keller, Christian Rieck, Christian Scheffer, and Arne Schmidt. Particle-based assembly using precise global control. In *Algorithms and Data Structures Symposium (WADS)*, pages 513–527, 2021. doi:10.1007/978-3-030-83508-8_37.

46 Jakob Keller, Christian Rieck, Christian Scheffer, and Arne Schmidt. Particle-based assembly using precise global control. *Algorithmica*, 84(10):2871–2897, 2022. A conference version of this paper appeared at WADS 2021 [45]. doi:10.1007/S00453-022-00992-2.

47 Stefan Kiefer and Andrew Ryzhikov. Efficiently computing the minimum rank of a matrix in a monoid of zero-one matrices. In *Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 61:1–61:22, 2025. doi:10.4230/LIPIcs.STACS.2025.61.

48 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.

49 Dexter Kozen. Lower bounds for natural proof systems. In *Symposium on Foundations of Computer Science*, pages 254–266, 1977. doi:10.1109/SFCS.1977.16.

50 Fabian C. Landers, Lukas Hertle, Vitaly Pustovalov, Derick Sivakumaran, Cagatay M. Oral, Oliver Brinkmann, Kirstin Meiners, Pascal Theiler, Valentin Gantenbein, Andrea Veciana, Michael Mattmann, Silas Riss, Simone Gervasoni, Christophe Chautems, Hao Ye, Semih Sevim, Andreas D. Flouris, Josep Puigmartí-Luis, Tiago Sotto Mayor, Pedro Alves, Tessa Lühmann, Xiangzhong Chen, Nicole Ochsenbein, Ueli Moehrlen, Tilman Schubert, Zsolt Kulcsar, Philipp Gruber, Miriam Weisskopf, Quentin Boehler, Salvador Pané, and Bradley J. Nelson. Clinically ready magnetic microrobots for targeted therapies. *Science*, 390(6774):710–715, 2025. doi:10.1126/science.adx1708.

51 Arun Mahadev, Dominik Krupke, Sándor P. Fekete, and Aaron T. Becker. Mapping and coverage with a particle swarm controlled by uniform inputs. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 1097–1104, 2017. doi:10.1109/IROS.2017.8202280.

52 Arun V. Mahadev, Dominik Krupke, Jan-Marc Reinhardt, Sándor P. Fekete, and Aaron T. Becker. Collecting a swarm in a grid environment using shared, global inputs. In *International Conference on Automation Science and Engineering (CASE)*, pages 1231–1236, 2016. doi:10.1109/COASE.2016.7743547.

53 Sheryl Manzoor, Samuel Sheckman, Jarrett Lonsford, Hoyeon Kim, Min Jun Kim, and Aaron T. Becker. Parallel self-assembly of polyominoes under uniform control inputs. *Robotics and Automation Letters*, 2(4):2040–2047, 2017. doi:10.1109/LRA.2017.2715402.

54 Rachel A. Moan, Victor M. Baez, Aaron T. Becker, and Jason M. O'Kane. Aggregation and localization of simple robots in curved environments. In *International Conference on Robotics and Automation (ICRA)*, pages 165–171, 2020. doi:10.1109/ICRA40945.2020.9197198.

55 Nils Morawietz, Carolin Rehs, and Mathias Weller. A timecop's work is harder than you think. In *Mathematical Foundations of Computer Science (MFCS)*, pages 71:1–71:14, 2020. doi:10.4230/LIPIcs.MFCS.2020.71.

56 Balas K. Natarajan. An algorithmic approach to the automated design of parts orienters. In *Symposium on Foundations of Computer Science*, pages 132–142, 1986. doi:10.1109/SFCS.1986.5.

57 Jason M. O'Kane and Steven M. LaValle. Localization with limited sensing. *Transactions on Robotics*, 23(4):704–716, 2007. doi:10.1109/TRO.2007.900636.

58 Kari-Jouko Räihä and Esko Ukkonen. The shortest common supersequence problem over binary alphabet is NP-complete. *Theoretical Computer Science*, 16:187–198, 1981. doi:10.1016/0304-3975(81)90075-X.

59 Igor K. Rystsov. Polynomial complete problems in automata theory. *Information Processing Letters*, 16(3):147–151, 1983. doi:10.1016/0020-0190(83)90067-4.

60 Sven Sandberg. Homing and synchronizing sequences. In *Model-Based Testing of Reactive Systems*, pages 5–33, 2004. doi:10.1007/11498490_2.

61 Walter J. Savitch. Deterministic simulation of non-deterministic turing machines (detailed abstract). In *Symposium on Theory of Computing (STOC)*, pages 247–248, 1969. doi:10.1145/800169.805439.

62 Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, 1970. A conference version of this paper appeared at STOC 1969 [61]. doi:10.1016/S0022-0000(70)80006-X.

63 Arne Schmidt, Sheryl Manzoor, Li Huang, Aaron T. Becker, and Sándor P. Fekete. Efficient parallel self-assembly under uniform control inputs. *Robotics and Automation Letters*, 3(4):3521–3528, 2018. doi:10.1109/LRA.2018.2853758.

64 Hamed Mohtasham Shad, Rose Morris-Wright, Erik D. Demaine, Sándor P. Fekete, and Aaron T. Becker. Particle computation: Device fan-out and binary memory. In *International Conference on Robotics and Automation (ICRA)*, pages 5384–5389, 2015. doi:10.1109/ICRA.2015.7139951.

65 Yaroslav Shitov. An improvement to a recent upper bound for synchronizing words of finite automata. *Journal of Automata, Languages and Combinatorics*, 24(2-4):367–373, 2019. doi:10.25596/JALC-2019-367.

66 Larry J. Stockmeyer and Albert R. Meyer. Word problems requiring exponential time: Preliminary report. In *Symposium on Theory of Computing (STOC)*, pages 1–9, 1973. doi:10.1145/800125.804029.

67 Marek Szykuła. Improving the upper bound on the length of the shortest reset word. In *Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 56:1–56:13, 2018. doi:10.4230/LIPIcs.STACS.2018.56.

68 Marek Szykuła and Adam Zyzik. An improved algorithm for finding the shortest synchronizing words. In *European Symposium on Algorithms (ESA)*, pages 85:1–85:15, 2022. doi:10.4230/LIPIcs.ESA.2022.85.

69 Mikhail V. Volkov. Synchronizing automata and the Černý conjecture. In *Language and Automata Theory and Applications (LATA)*, pages 11–27, 2008. doi:10.1007/978-3-540-88282-4_4.

70 Mikhail V. Volkov. Synchronization of finite automata. *Russian Mathematical Surveys*, 77(5):819–891, 2022. doi:10.4213/rm10005e.

71 Vojtěch Vorel. Subset synchronization of transitive automata. In *Automata and Formal Languages (AFL)*, pages 370–381, 2014. doi:10.4204/EPTCS.151.26.

72 Vojtěch Vorel. Subset synchronization and careful synchronization of binary finite automata. *International Journal of Foundations of Computer Science*, 27(5):557–578, 2016. A conference version of this paper appeared at AFL 2014 [71]. doi:10.1142/S0129054116500167.

73 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.