Efficient Reconfiguration of Tile Arrangements by a Single Active Robot

Aaron T. Becker* Sándor P. Fekete† Jonas Friemel‡ Ramin Kosfeld† Peter Kramer† Harm Kube§ Christian Rieck¶ Christian Scheffer‡ Arne Schmidt†

Abstract

We consider the problem of reconfiguring a two-dimensional connected grid arrangement of passive building blocks from a start configuration to a goal configuration, using a single active robot that can move on the tiles, remove individual tiles from a given location and physically move them to a new position by walking on the remaining configuration. The objective is to determine a schedule that minimizes the overall makespan, while keeping the tile configuration connected. We provide both negative and positive results. (1) We generalize the problem by introducing weighted movement costs, which can vary depending on whether tiles are carried or not, and prove that this variant is NP-hard. (2) We give a polynomial-time constant-factor approximation algorithm for the case of disjoint start and target bounding boxes, which additionally yields optimal carry distance for 2-scaled instances.

1 Introduction

Building and modifying structures consisting of many basic components is an important objective, both in fundamental theory and in a spectrum of practical settings. Transforming such structures with the help of autonomous robots is particularly relevant in very large [17] and very small dimensions [41] that are hard to access for direct human manipulation, e.g., in extraterrestrial space [10,34] or in microscopic environments [7]. This gives rise to the natural algorithmic problem of rearranging a given start configuration of many *passive* objects by a small set of *active* agents to a desired target configuration. Performing such reconfiguration at scale faces a number of critical challenges, including (i) the cost and weight of materials, (ii) the potentially disastrous accumulation of errors, (iii) the development of simple yet resilient agents to carry out the active role, and (iv) achieving overall feasibility and efficiency.

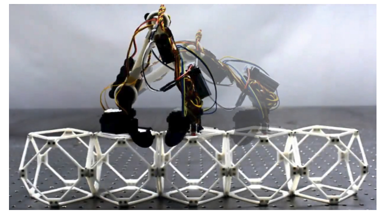


Figure 1: A Bill-E robot moves on a configuration of light-weight material and relocate individual voxels for overall reconfiguration. Photo adapted from [32].

In recent years, significant advances have been made to deal with these difficulties. Macroscopically, ultra-light and scalable composite lattice materials [13,28,29] tackle the first problem, making it possible to construct modular, reconfigurable structures with platforms such as NASA's Bill-E and Armadas [28,31,33]; the underlying self-adjusting lattice also resolves the issues of accuracy and error correction (ii), allowing it to focus on discrete, combinatorial structures, consisting of regular tiles (in two dimensions) or voxels (in three dimensions). A further step has been the development of simple autonomous robots [11,32] that can be used to carry out complex tasks (iii), as shown in Figure 1: The robot can move on the tile arrangement, remove individual tiles and physically relocate them to a new position by walking on the remaining configuration, which needs to remain connected at all times. At microscopic scales, advances in micro- and nanobots [39,40] offer novel ways to (re)configure objects and mechanisms, e.g., assembling specific structures or gathering in designated locations for tasks like targeted drug delivery [9,35].

In this paper, we deal with challenge (iv): How can we use such a robot to transform a given start configuration into a desired goal arrangement, as quickly as possible?

1.1 Our contributions

We investigate the problem of finding minimum cost reconfiguration schedules for a single active robot operating on a (potentially large) number of tiles, and give the following results. Details for statements marked by ($\star$) can be found in the full version of our paper [8].

*Department of Electrical Engineering, University of Houston, USA, atbecker@uh.edu

†Department of Computer Science, TU Braunschweig, Germany, {kosfeld, kramer, aschmidt}@ibr.cs.tu-bs.de, s.fekete@tu-bs.de

‡Department of Electrical Engineering and Computer Science, Bochum University of Applied Sciences, Germany, {jonas.friemel, christian.scheffer}@hs-bochum.de

§Department of Computer Science, TU Berlin, Germany, h.kube@campus.tu-berlin.de

¶Department of Discrete Mathematics, University of Kassel, Germany, christian.rieck@mathematik.uni-kassel.de

(1) We present a generalized version of the problem, parameterized by movement costs, which can vary depending on whether tiles are carried or not, and show that this is NP-hard.

(2) We give a polynomial-time constant-factor approximation algorithm for the case of disjoint start and target bounding boxes. Our approach further yields optimal carry distance for 2-scaled instances.

1.2 Related work

Recently, the authors of [24,25] showed that computing optimal schedules for BILL-E bots, see [31] and Figure 1, is NP-hard in unweighted models. They designed a heuristic approach that exploits rapidly exploring random trees (RRT) and a time-dependent variant of the A* algorithm, as well as target swapping heuristics to reduce the overall distance traveled for multiple robots. The authors of [12] present an algorithm for computing feasible build orders that adhere to three-dimensional tile placement constraints in the ARMADAS project.

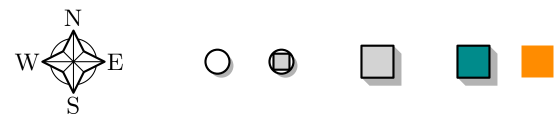
A different context for reconfiguration arises from programmable matter [15]. Here, finite automata are capable of building bounding boxes of tiles around polyominoes, as well as scale and rotate them while maintaining connectivity at all times [20,38]. On hexagonal grids, finite automata can build and recognize simple shapes such as triangles or parallelograms [26, 27, 30] as well as more complicated shapes if they are able to recognize nodes that belong to the target shape [23].

When considering active matter, arrangements composed of self-moving objects (or agents), numerous models exist [5,6,14,18,37]. For example, in the *sliding cube model* (or the *sliding square model* in two dimensions) first introduced in 2003 [21, 22], agents are allowed to slide along other, temporarily static, agents for reconfiguration, but must maintain connectivity throughout. The authors of [3] show that sequential reconfiguration in two dimensions is always possible, but deciding the minimal makespan is NP-complete. Recent work presents similar results for the three-dimensional setting [1,36], as well as parallel movement [4].

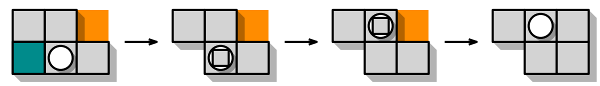
In a relaxed model, the authors of [18, 19] show that parallel connected reconfiguration of swarms of (labeled) agents is NP-complete, even for makespan 2, and present algorithms for schedules with constant *stretch*; the ratio of a schedule's makespan to the individual maximum minimum distance between start and target.

1.3 Preliminaries

For the following, we refer to Figure 2. We are given a fixed set of n indistinguishable square *tile* modules located at discrete, unique positions (x, y) in the infinite integer grid $\mathbb{Z}^2$. If their positions induce a connected subgraph of the grid, where two positions are connected if either their x- or y-coordinate differs by 1, we say that the tiles form a connected *configuration* or *polyomino*. Let $\mathcal{C}(n)$ refer to the set of all polyominoes of n tiles.



(a) Left to right: The cardinal directions, a robot without and with payload, a tile, and tiles that only exist in either the start or the target configuration, respectively.



(b) A schedule for the instance on the left. The robot moves on tiles, picking up and dropping off adjacent tiles.

Figure 2: A brief overview of legal operations.

Consider a robot that occupies a single tile at any given time. In discrete time steps, the robot can either move to an adjacent tile, pick up a tile from an adjacent position (if it is not already carrying one), or place a carried tile at an adjacent unoccupied position. Tiles can be picked up only if connectivity is preserved.

We use cardinal directions; the unit vectors $(1, 0)$ and $(0, 1)$ correspond to *east* and *north*, respectively. Naturally, their opposing vectors extend *west* and *south*.

A *schedule* S is a finite, connectivity-preserving sequence of operations to be performed by the robot. As the robot's motion is restricted to movements on the polyomino, we refer to distances it travels as *geodesic* distances. Let $d_C(S)$ denote the *carry distance*, which is the number of robot moves while carrying a tile (i.e., the sum of geodesic distances between consecutive pickups and drop-offs), plus the number of all pickups and drop-offs in S. Accounting for the remaining moves without carrying a tile, the *empty distance* $d_E(S)$ is the sum of geodesic distances between drop-offs and pickups in S. For $C_s, C_t \in \mathcal{C}(n)$, we say that S is a schedule for $C_s \rightrightarrows C_t$ exactly if it transforms C_s into C_t.

Problem statement. We consider SINGLE ROBOT RECONFIGURATION: Given configurations $C_s, C_t \in \mathcal{C}(n)$ and a rational weight factor $\lambda \in [0, 1]$, determine a schedule S for $C_s \rightrightarrows C_t$ that minimizes the *weighted makespan* $|S| := \lambda \cdot d_E(S) + d_C(S)$. We refer to the minimum weighted makespan for a given instance as OPT.

2 Computational complexity of the problem

We investigate the computational complexity of the decision variant of the reconfiguration problem. In particular, we prove that the problem is NP-hard for any rational $\lambda \in [0, 1]$. This generalizes the previous result from [25], which handles the case of $\lambda = 1$.

Theorem 1 Single Robot Reconfiguration *is* NP*-hard for any rational* λ*.*

We distinguish between the two cases of (1) $\lambda \in (0, 1]$ and (2) $\lambda = 0$. For (1), we modify the construction from [25] to yield a slightly stronger statement, reducing from the Hamiltonian path problem in grid graphs. The primary result of our reduction is the following.

Lemma 2 ($\star$) Single Robot Reconfiguration *is* NP*-hard for* $\lambda \in (0, 1]$*.*

Our modified construction further implies that it is NP-hard to decide the minimal number of pickup/drop-off operations in an optimal schedule with $\lambda \in (0, 1]$.

Corollary 3 *Given two configurations* $C_s, C_t \in \mathcal{C}(n)$ *and an integer* $k \in \mathbb{N}$*, it is* NP*-hard to decide whether there exists an optimal schedule* $C_s \rightrightarrows C_t$ *with at most* k *pickup (at most* k *drop-off) operations, if* $\lambda \in (0, 1]$*.*

The reduction for (2) is significantly more involved: As $\lambda = 0$, the robot is effectively allowed to "teleport" across the configuration, albeit only without cargo.

Lemma 4 ($\star$) Single Robot Reconfiguration *is* NP*-hard for* $\lambda = 0$*.*

We reduce from Planar Monotone 3Sat [16], following ideas by the authors of [3] for the sequential sliding square problem. This variant of the SAT problem asks whether a given Boolean formula φ in conjunctive normal form is satisfiable, given the following properties: First, each clause consists of at most 3 literals, all either positive or negative. Second, the clause-variable incidence graph G_φ admits a plane drawing in which variables are placed on the x-axis, and all positive (resp., negative) clauses are located in the same half-plane, such that edges do not cross the x-axis, see Figure 3. We assume, without loss of generality, that each clause

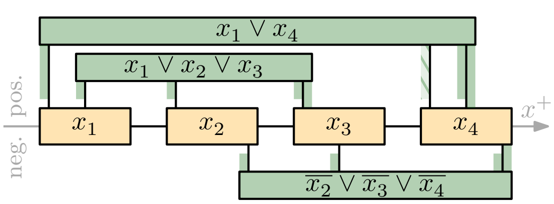


Figure 3: The embedded clause-variable incidence graph of $\varphi = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_4) \wedge (\overline{x_2} \vee \overline{x_3} \vee \overline{x_4})$.

contains exactly three literals; otherwise, we extend a shorter clause with a redundant copy of one of the existing literals, e.g., $(x_1 \vee x_4)$ becomes $(x_1 \vee x_4 \vee x_4)$.

Our reduction maps from an instance φ of Planar Monotone 3Sat to an instance $\mathcal{I}_\varphi$ of Single Robot Reconfiguration such that the minimal feasible makespan $\mathcal{I}_\varphi$ is determined by whether φ is satisfiable. Recall that, due to $\lambda = 0$, we only account for carry distance. Consider an embedding of the clause-variable incidence graph as above, where C and V refer to the m clauses and n variables of φ, respectively.

We construct $\mathcal{I}_\varphi$ as follows. A *variable gadget* is placed on the x-axis for each $x_i \in V$, and connected along the axis in a straight line. Intuitively, the variable gadget asks the robot to move a specific tile west along one of two feasible paths, which encode a value assignment. These paths are highlighted in red and blue in Figure 4, each of length exactly $9(\delta(x_i) + 1)$, where $\delta(x_i)$ refers to the degree of x_i in the clause-variable incidence graph. Both paths require the robot to place

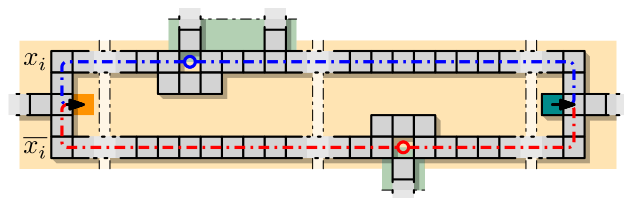


Figure 4: The variable gadget; the middle segments are repeated to match the incident clauses.

its payload into the highlighted gaps to minimize the makespan, stepping over it before picking up again. We further add one *clause gadget* per clause, connected to its incident variables. These are effectively combs with three or four prongs, connected to a variable gadget by the rightmost one, see Figure 5. The remaining prongs

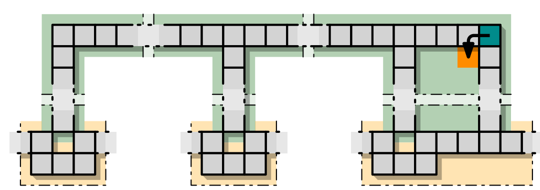


Figure 5: A clause gadget and its variable attachment.

each terminate at a position diagonally adjacent to the variable gadget of a corresponding literal. To solve the gadget, the rightmost prong must be temporarily disconnected from the comb, meaning that another prong must have been connected to a variable gadget first. Such a link can be established without extra cost if one of the adjacent variable gadgets has a matching Boolean value, allowing us to use its payload to temporarily close one of the gaps marked by circles in Figure 4.

We can show that a weighted makespan of $29m + 9n$ can be achieved exactly if φ is satisfiable; recall that m and n denote the number of clauses and variables in φ, respectively. Theorem 1 is then simply the union of Lemmas 2 and 4.

3 Constant-factor approximation for 2-scaled instances

We now turn to the case in which the configurations have *disjoint bounding boxes*, i.e., there exists an axis-parallel bisector that separates the configurations. Without loss of generality, let this bisector be horizontal such that the target configuration lies south. We present a constant-factor approximation algorithm. For the remainder of this section, we additionally impose the constraint that both the start and target configurations are 2-*scaled*, i.e., they consist of 2×2-squares of tiles aligned with a corresponding grid. In Section 4, we extend our result to non-scaled configurations.

Theorem 5 *For any $\lambda \in [0,1]$, there exists a constant c such that for any pair of 2-scaled configurations $C_s, C_t \in \mathcal{C}(n)$ with disjoint bounding boxes, we can efficiently compute a schedule for $C_s \rightrightarrows C_t$ with weighted makespan at most $c \cdot \mathsf{OPT}$.*

Our algorithm utilizes *histograms* as intermediate configurations. A histogram consists of a *base* strip of unit height (a single tile, also when 2-scaled) and unit width *columns* attached orthogonally to its base. The direction of its columns determines the orientation of a histogram, e.g., H_s in Figure 6 is *north-facing*.

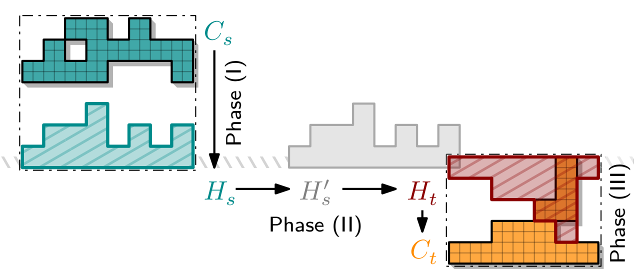


Figure 6: An example for a start and target configuration C_s, C_t, the intermediate histograms H_s, H_t with a shared baseline, and the horizontally translated H'_s that shares a tile with H_t. If H_s and H_t overlap, $H'_s := H_s$.

As illustrated in Figure 6, we proceed in three phases.

Phase (I). Iteratively move components of C_s south until it forms a north-facing histogram H_s such that its base shares its y-coordinate with a module of C_t.

Phase (II). Translate H_s to overlap with the target bounding box and transform it into a specific south-facing histogram H_t within the bounding box of C_t.

Phase (III). Finally, apply Phase (I) in reverse to obtain C_t from H_t.

We reduce this to two subroutines: Transforming any 2-scaled configuration into a histogram and reconfiguring any two histograms into one another. Note that while the respectively obtained histograms are 2-scaled when the initial configurations are, the reconfiguration between two opposite-facing histograms actually does not depend on this condition. In fact, the same reconfiguration routines remain valid even when the histograms involved are not 2-scaled, highlighting the broader applicability of our method.

3.1 Preliminaries for the algorithm

For our algorithm, we use the following terms. Given two configurations $C_s, C_t \in \mathcal{C}(n)$, the weighted bipartite graph $G_{C_s,C_t} = (C_s \cup C_t, C_s \times C_t, L_1)$ assigns each edge a weight equal to the L_1-distance between its end points.

A *perfect matching* M in G_{C_s,C_t} is a subset of edges such that every vertex is incident to exactly one of them; its weight $w(M)$ is defined as the sum of its edge weights. By definition, there exists at least one such matching in G_{C_s,C_t}. Furthermore, a *minimum weight perfect matching* (MWPM) is a perfect matching M of minimum weight $\sigma(C_s, C_t) = w(M)$, which is a natural lower bound on the necessary carry distance, i.e., OPT.

Let S be any schedule for $C_s \rightrightarrows C_t$. Then S induces a perfect matching in G_{C_s,C_t}, as it moves every tile of C_s to a distinct position of C_t. We say that S has *optimal carry distance* exactly if $d_C(S) = \sigma(C_s, C_t)$.

3.2 Phase (I): Transforming into a histogram

We proceed by constructing a histogram from an arbitrary 2-scaled configuration by moving tiles strictly in one cardinal direction.

Lemma 6 ($\star$) *Let $C_s \in \mathcal{C}(n)$ be a 2-scaled polyomino and let H_s be a histogram that can be created from C_s by moving tiles in only one cardinal direction. We can efficiently compute a schedule with optimal carry distance and total makespan $\mathcal{O}(n + \sigma(C_s, H_s))$ for $C_s \rightrightarrows H_s$.*

To achieve this, we iteratively move sets of tiles in the target direction by two units, until the histogram is constructed. We give a high-level explanation of our approach by example of a north-facing histogram, as depicted in Figure 7.

Let P be any intermediate 2-scaled polyomino obtained by moving tiles south while realizing $C_s \rightrightarrows H_s$. Let H be the set of maximal vertical strips of tiles that contain a base tile in H_s, i.e., all tiles that do not need to be moved further south. We define the *free components* of P as the set of connected components in $P \setminus H$. By definition, these exist exactly if P is not equivalent to H_s, and once a tile in P becomes part of H, it is not moved again until H_s is obtained.

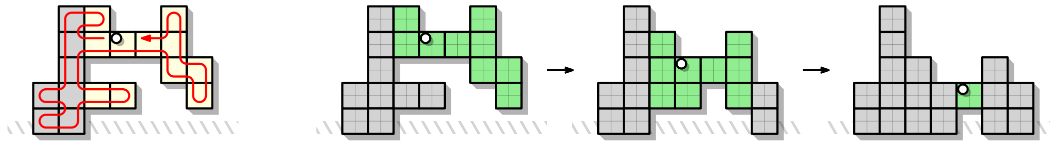


Figure 7: The figure visualizes Lemma 6. Left: A walk across all tiles (red), the set H (gray) and two free components (yellow). Right: Based on the walk, free components are iteratively moved south to reach a histogram shape. The free component that is going to be translated next is highlighted in green.

We compute a *walk* that covers the polyomino, i.e., a path that is allowed to traverse edges multiple times and visits each vertex at least once, by depth-first traversal on an arbitrary spanning tree of P. The robot then continuously follows this walk, iteratively moving free components south and updating its path accordingly: Whenever it enters a free component F of P, it performs a subroutine with makespan $\mathcal{O}(|F|)$ that moves F south by two units. Adjusting the walk afterward may increase its length by $\mathcal{O}(|F|)$ units per free component.

Upon completion of this algorithm, we can bound both the total time spent performing the subroutine and the additional cost incurred by extending the walk by $\mathcal{O}(\sigma(C_s, H_s))$. Taking into account the initial length of the walk, the resulting makespan is $\mathcal{O}(n + \sigma(C_s, H_s))$. The subroutine for the translation of free components can be stated as follows.

Lemma 7 *We can efficiently compute a schedule of makespan $\mathcal{O}(|F|)$ to translate any free component F of a 2-scaled polyomino in the target direction by two units.*

Proof. Without loss of generality, let the target direction be south. We show how to translate F south by one unit without losing connectivity, which we do twice.

We follow a bounded-length walk across F that visits exclusively tiles with a tile neighbor in northern direction. Such a walk can be computed by depth-first traversal of F. Whenever the robot enters a maximal vertical strip of F for the first time, it picks up the northernmost tile, places it at the first unoccupied position to its south, and continues its traversal.

As each strip is handled exactly once, the total movement cost on vertical strips for carrying tiles and returning to the pre-pickup position is bounded by $\mathcal{O}(|F|)$. This bound also holds for the length of the walk. □

Applying Lemma 7 to the entire polyomino, rather than restricting it to a free component, allows the translation of the polyomino in any direction with asymptotically optimal makespan.

Corollary 8 *Any 2-scaled polyomino can be translated by k units in any cardinal direction by a schedule of weighted makespan $\mathcal{O}(n \cdot k)$.*

3.3 Phase (II): Reconfiguring histograms

It remains to show how to transform one histogram into the other. By the assumption of the existence of a horizontal bisector between the bounding boxes of C_s and C_t, the histogram H_s is north-facing, whereas H_t is south-facing. The bounding box of C_s is vertically extended to share exactly one y-coordinate with the bounding box of C_t, and this is where both histogram bases are placed; see Figure 6 for an illustration. Note that the histograms may not yet overlap. However, by Corollary 8, the tiles in H_s can be moved toward H_t with asymptotically optimal makespan until both histogram bases share a tile.

Lemma 9 ($\star$) *Let H_s and H_t be opposite-facing histograms that share a base tile. We can efficiently compute a schedule of makespan $\mathcal{O}(n + \sigma(H_s, H_t))$ for $H_s \rightrightarrows H_t$ with optimal carry distance.*

As illustrated on the left side of Figure 8, we order all tiles to be moved, as well as all positions to be populated, from west to east and north to south. By that, iteratively moving the first remaining tile from $H_s \setminus H_t$ to $H_t \setminus H_s$ yields a schedule with the desired properties. The right side of Figure 8 depicts a special case in which the westernmost positions in $H_t \setminus H_s$ cannot be reached initially. This case requires a preprocessing in which the baseline is first extended to the west.

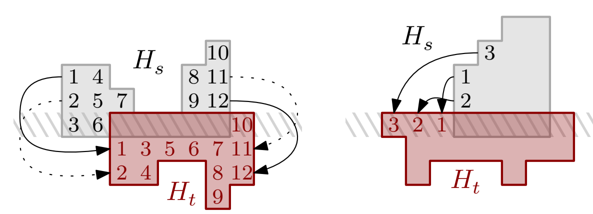


Figure 8: Left: Ordering of tile moves for $H_s \rightrightarrows H_t$. Right: If the westernmost unoccupied position in H_t is unreachable, the base may need to be extended first.

3.4 Correctness of the algorithm

In the previous sections, we presented schedules for each phase of the overall algorithm. We will now leverage these insights to prove the main result of this section, restated here.

Theorem 5 *For any $\lambda \in [0,1]$, there exists a constant c such that for any pair of 2-scaled configurations $C_s, C_t \in \mathcal{C}(n)$ with disjoint bounding boxes, we can efficiently compute a schedule for $C_s \rightrightarrows C_t$ with weighted makespan at most $c \cdot \mathit{OPT}$.*

Proof. By Lemmas 6 and 9, the makespan of the three phases is bounded by, respectively,

$$\mathcal{O}(n+\sigma(C_s,H_s)),\ \mathcal{O}(n+\sigma(H_s,H_t)),\ \mathcal{O}(n+\sigma(H_t,C_t)),$$

which we now bound by $\mathcal{O}(\sigma(C_s,C_t))$, proving asymptotic optimality for $C_s \rightrightarrows C_t$.

Clearly, $n \in \mathcal{O}(\sigma(C_s,C_t))$, as each of the n tiles has to be moved due to $C_s \cap C_t = \varnothing$. We prove a tight bound on the remaining terms, i.e.,

$$\sigma(C_s,H_s)+\sigma(H_s,H_t)+\sigma(H_t,C_t)=\sigma(C_s,C_t). \quad (\blacklozenge)$$

In Phase (I), tiles are moved exclusively toward the bounding box of C_t along shortest paths to obtain H_s; therefore, $\sigma(C_s,C_t)=\sigma(C_s,H_s)+\sigma(H_s,C_t)$. The same applies to the reverse of Phase (III), i.e., $\sigma(C_t,H_s)=\sigma(C_t,H_t)+\sigma(H_t,H_s)$. As the lower bound is symmetric, Equation ($\blacklozenge$) immediately follows. The total makespan is thus in $\mathcal{O}(\sigma(C_s,C_t))=\mathcal{O}(\mathsf{OPT})$. □

Lemmas 6 and 9 move tiles on shortest paths and establish schedules that minimize the carry distance. Equation ($\blacklozenge$) ensures that the combined paths remain shortest possible with regard to $C_s \rightrightarrows C_t$. Thus, the provided schedule has optimal carry distance. In particular, we obtain an optimal schedule when $\lambda = 0$, which corresponds to the case where the robot incurs no cost for movement when not carrying a tile.

Corollary 10 *An optimal schedule for $C_s \rightrightarrows C_t$ can be computed efficiently for any two 2-scaled configurations $C_s, C_t \in \mathcal{C}(n)$ with disjoint bounding boxes and $\lambda = 0$.*

4 Constant-factor approximation for general instances

The key advantage of 2-scaled instances is the absence of cut vertices, which simplifies the maintenance of connectivity during reconfiguration. Therefore, the challenge with general instances lies in managing cut vertices.

Most parts of our previous method already work independent of the configuration scale. The only modification required concerns Lemma 7, as the polyomino may become disconnected while moving free components that are not 2-scaled. To preserve local connectivity during the reconfiguration, we utilize two auxiliary tiles as a patching mechanism; this technique is also employed in other models [2, 37].

Lemma 11 ($\star$) *Let the robot hold two auxiliary tiles. Given a free component F on a polyomino P, we can efficiently compute a schedule of makespan $\mathcal{O}(|F|)$ to translate F in the target direction by one unit.*

A key idea in the proof of Lemma 11 is partitioning the configuration into horizontal and vertical strips.

By using auxiliary tiles, we are able to preserve connectivity while translating horizontal strips; see Figure 9. To guarantee connectivity, the strips must be moved in a specific order; we resolve this constraint using a recursive strategy that systematically moves dependent strips in the correct sequence.

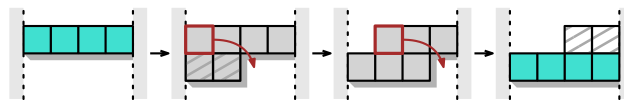


Figure 9: Using two auxiliary tiles to retain connectivity while moving a unit-height strip south.

Theorem 12 ($\star$) *For any $\lambda \in [0,1]$, there exists a constant c such that for any two configurations $C_s, C_t \in \mathcal{C}(n)$ with disjoint bounding boxes, we can efficiently compute a schedule for $C_s \rightrightarrows C_t$ with weighted makespan at most $c \cdot \mathit{OPT}$.*

To prove Theorem 12, it suffices to demonstrate how the auxiliary tiles can be obtained (any two tiles that do not break connectivity can serve this purpose) and that carrying two tiles can be emulated by sequentially carrying one tile after the other.

5 Discussion

We presented progress on the reconfiguration problem for tile-based structures (i.e., polyominoes) within abstract material-robot systems. In particular, we showed that the problem is NP-hard for any weighting between moving with or without carrying a tile.

Complementary to this negative result, we developed an algorithm to reconfigure two polyominoes into one another in the case that both configurations are contained in disjoint bounding boxes. The computed schedules are within a constant factor of the optimal reconfiguration schedule. It is easy to see that our approach can also be used to construct a polyomino, rather than reconfigure one into another. Instead of deconstructing a start configuration to generate building material, we can assume that tiles are located within a "depot" from which they can be picked up. Performing the second half of the algorithm works as before and builds the target configuration out of tiles from the depot.

Several open questions remain. A natural open problem is to adapt the approach to instances in which the bounding boxes of the configurations intersect, i.e., they overlap, or are nested. This hinges on proving a good lower bound on the makespan of any such schedule. Note that a minimum-weight perfect matching does not provide a reliable lower bound in this setting, as the cost of an optimal solution can be arbitrarily larger; particularly in cases involving small matchings, as visualized in Figure 10.

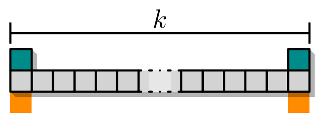


Figure 10: Although the MWPM has weight 4, the robot is required to perform at least $k-1 \gg 4$ empty moves. This discrepancy shows that the MWPM significantly underestimates the true cost, making it unsuitable as a lower bound for an approximation in the setting where bounding boxes intersect.

However, an alternative would be to provide an algorithm that achieves worst-case optimality. Furthermore, the more general question about three dimensional settings remain open. Our methods could likely be generalized for parallel execution by multiple robots. More intricate is the question on whether a fully distributed approach is possible.

Acknowledgements

This work was partially supported by multiple sources: German Research Foundation (DFG) project "Space Ants" (FE 407/22-1 and SCHE 1931/4-1) and grant 522790373, as well as by grants NSF IIS-2130793, ARL W911NF-23-2-0014, and DAF AFX23D-TCSO1.

References

[1] Zachary Abel, Hugo A. Akitaya, Scott Duke Kominers, Matias Korman, and Frederick Stock. A universal in-place reconfiguration algorithm for sliding cube-shaped robots in a quadratic number of moves. In *Symposium on Computational Geometry (SoCG)*, pages 1:1–1:14, 2024. doi:10.4230/LIPICS.SOCG.2024.1.

[2] Hugo A. Akitaya, Esther M. Arkin, Mirela Damian, Erik D. Demaine, Vida Dujmovic, Robin Y. Flatland, Matias Korman, Belén Palop, Irene Parada, André van Renssen, and Vera Sacristán. Universal reconfiguration of facet-connected modular robots by pivots: The $O(1)$ musketeers. *Algorithmica*, 83:1316–1351, 2021. doi:10.1007/s00453-020-00784-6.

[3] Hugo A. Akitaya, Erik D. Demaine, Matias Korman, Irina Kostitsyna, Irene Parada, Willem Sonke, Bettina Speckmann, Ryuhei Uehara, and Jules Wulms. Compacting squares: Input-sensitive in-place reconfiguration of sliding squares. In *Scandinavian Symposium on Algorithm Theory (SWAT)*, pages 4:1–4:19, 2022. doi:10.4230/LIPICS.SWAT.2022.4.

[4] Hugo A. Akitaya, Sándor P. Fekete, Peter Kramer, Saba Molaei, Christian Rieck, Frederick Stock, and Tobias Wallner. Sliding squares in parallel. In *European Symposium on Algorithms (ESA)*, 2025. doi:10.48550/arXiv.2412.05523.

[5] Abdullah Almethen, Othon Michail, and Igor Potapov. Pushing lines helps: Efficient universal centralised transformations for programmable matter. *Theoretical Computer Science*, 830:43–59, 2020. doi:10.1016/j.tcs.2020.04.026.

[6] Abdullah Almethen, Othon Michail, and Igor Potapov. On efficient connectivity-preserving transformations in a grid. *Theoretical Computer Science*, 898:132–148, 2022. doi:10.1016/j.tcs.2022.09.016.

[7] 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.

[8] Aaron T. Becker, Sándor P. Fekete, Jonas Friemel, Ramin Kosfeld, Peter Kramer, Harm Kube, Christian Rieck, Christian Scheffer, and Arne Schmidt. Efficient reconfiguration of tile arrangements by a single active robot, 2025. arXiv:2502.09299.

[9] 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.

[10] Mohamed Khalil Ben-Larbi, Kattia Flores Pozo, Tom Haylok, Mirue Choi, Benjamin Grzesik, Andreas Haas, Dominik Krupke, Harald Konstanski, Volker Schaus, Sándor P. Fekete, Christian Schurig, and Enrico Stoll. Towards the automated operations of large distributed satellite systems. Part 1: Review and paradigm shifts. *Advances in Space Research*, 67(11):3598–3619, 2021. doi:10.1016/j.asr.2020.08.009.

[11] Christopher G. Cameron, Zach Fredin, and Neil Gershenfeld. Discrete assembly of unmanned aerial systems. In *International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 339–344, 2022. doi:10.1109/ICUAS54217.2022.9836082.

[12] Kenneth Cheung, Irina Kostitsyna, and Tom Peters. Assembly order planning for modular structures by autonomous multi-robot systems. In *International Conference on Robotics and Automation (ICRA)*, 2025.

[13] Kenneth C. Cheung and Neil Gershenfeld. Reversibly assembled cellular composite materials. *Science*, 341(6151):1219–1221, 2013. doi:10.1126/science.1240889.

[14] Matthew Connor and Othon Michail. Transformation of modular robots by rotation: $3 + 1$ musketeers for all orthogonally convex shapes. *Journal of Computer and System Sciences*, 150:103618, 2025. doi:10.1016/j.jcss.2024.103618.

[15] Joshua J. Daymude, Kristian Hinnenthal, Andréa W. Richa, and Christian Scheideler. Computing by programmable particles. *Distributed Computing by Mobile Entities: Current Research in Moving and Computing*, pages 615–681, 2019. doi:10.1007/978-3-030-11072-7_22.

[16] Mark de Berg and Amirali Khosravi. Optimal binary space partitions for segments in the plane. *International Journal on Computational Geometry and Applications*, 22(3):187–206, 2012. doi:10.1142/S0218195912500045.

[17] Sándor P. Fekete. Coordinating swarms of objects at extreme dimensions. In *International Workshop on Combinatorial Algorithms (IWOCA)*, pages 3–13, 2020. doi:10.1007/978-3-030-48966-3_1.

[18] Sándor P. Fekete, Phillip Keldenich, Ramin Kosfeld, Christian Rieck, and Christian Scheffer. Connected coordinated motion planning with bounded stretch. *Autonomous Agents and Multi-Agent Systems*, 37(2), 2023. doi:10.1007/S10458-023-09626-5.

[19] Sándor P. Fekete, Peter Kramer, Christian Rieck, Christian Scheffer, and Arne Schmidt. Efficiently reconfiguring a connected swarm of labeled robots. *Autonomous Agents and Multi-Agent Systems*, 38(2), 2024. doi:10.1007/s10458-024-09668-3.

[20] Sándor P. Fekete, Eike Niehs, Christian Scheffer, and Arne Schmidt. Connected reconfiguration of lattice-based cellular structures by finite-memory robots. *Algorithmica*, 84(10):2954–2986, 2022. doi:10.1007/s00453-022-00995-z.

[21] Robert Fitch, Zack J. Butler, and Daniela Rus. Reconfiguration planning for heterogeneous self-reconfiguring robots. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 2460–2467, 2003. doi:10.1109/IROS.2003.1249239.

[22] Robert Fitch, Zack J. Butler, and Daniela Rus. Reconfiguration planning among obstacles for heterogeneous self-reconfiguring robots. In *International Conference on Robotics and Automation (ICRA)*, pages 117–124, 2005. doi:10.1109/ROBOT.2005.1570106.

[23] Jonas Friemel, David Liedtke, and Christian Scheffer. Efficient shape reconfiguration by hybrid programmable matter. In *European Workshop on Computational Geometry (EuroCG)*, pages 14:1–14:8, 2025. arXiv:2501.08663.

[24] Javier Garcia, Michael Yannuzzi, Peter Kramer, Christian Rieck, and Aaron T. Becker. Connected reconfiguration of polyominoes amid obstacles using RRT*. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 6554–6560, 2022. doi:10.1109/IROS47612.2022.9981184.

[25] Javier Garcia, Michael Yannuzzi, Peter Kramer, Christian Rieck, Sándor P. Fekete, and Aaron T. Becker. Reconfiguration of a 2D structure using spatio-temporal planning and load transferring. In *International Conference on Robotics and Automation (ICRA)*, pages 8735–8741, 2024. doi:10.1109/ICRA57147.2024.10611057.

[26] Robert Gmyr, Kristian Hinnenthal, Irina Kostitsyna, Fabian Kuhn, Dorian Rudolph, and Christian Scheideler. Shape recognition by a finite automaton robot. In *International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 52:1–52:15, 2018. doi:10.4230/LIPIcs.MFCS.2018.52.

[27] Robert Gmyr, Kristian Hinnenthal, Irina Kostitsyna, Fabian Kuhn, Dorian Rudolph, Christian Scheideler, and Thim Strothmann. Forming tile shapes with simple robots. *Natural Computing*, 19(2):375–390, 2020. doi:10.1007/s11047-019-09774-2.

[28] Christine E. Gregg, Damiana Catanoso, Olivia Irene B. Formoso, Irina Kostitsyna, Megan E. Ochalek, Taiwo J. Olatunde, In Won Park, Frank M. Sebastianelli, Elizabeth M. Taylor, Greenfield T. Trinh, and Kenneth C. Cheung. Ultralight, strong, and self-reprogrammable mechanical metamaterials. *Science Robotics*, 9(86), 2024. doi:10.1126/scirobotics.adi2746.

[29] Christine E. Gregg, Joseph H. Kim, and Kenneth C. Cheung. Ultra-light and scalable composite lattice materials. *Advanced Engineering Materials*, 20(9):1800213, 2018. doi:10.1002/adem.201800213.

[30] Kristian Hinnenthal, David Liedtke, and Christian Scheideler. Efficient shape formation by 3D hybrid programmable matter: An algorithm for low diameter intermediate structures. In *Symposium on Algorithmic Foundations of Dynamic Networks (SAND)*, pages 15:1–15:20, 2024. doi:10.4230/LIPICS.SAND.2024.15.

[31] Ben Jenett and Kenneth Cheung. BILL-E: Robotic platform for locomotion and manipulation of lightweight space structures. In *Adaptive Structures Conference (ASC)*, 2017. doi:10.2514/6.2017-1876.

[32] Benjamin Jenett, Amira Abdel-Rahman, Kenneth Cheung, and Neil Gershenfeld. Material–robot system for assembly of discrete cellular structures. *IEEE Robotics and Automation Letters*, 4(4):4019–4026, 2019. doi:10.1109/LRA.2019.2930486.

[33] Benjamin Jenett, Daniel Cellucci, Christine Gregg, and Kenneth Cheung. Meso-scale digital materials: modular, reconfigurable, lattice-based structures. In *International Manufacturing Science and Engineering Conference (MSEC)*, 2016. doi:10.1115/MSEC2016-8767.

[34] Benjamin Jenett, Christine Gregg, Daniel Cellucci, and Kenneth Cheung. Design of multifunctional hierarchical space structures. In *Aerospace Conference*, pages 1–10, 2017. doi:10.1109/AERO.2017.7943913.

[35] 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.

[36] Irina Kostitsyna, Tim Ophelders, Irene Parada, Tom Peters, Willem Sonke, and Bettina Speckmann. Optimal in-place compaction of sliding cubes. In *Scandinavian Symposium on Algorithm Theory (SWAT)*, pages 31:1–31:14, 2024. doi:10.4230/LIPICS.SWAT.2024.31.

[37] Othon Michail, George Skretas, and Paul G. Spirakis. On the transformation capability of feasible mechanisms for programmable matter. *Journal of Computer and System Sciences*, 102:18–39, 2019. doi:10.1016/j.jcss.2018.12.001.

[38] Eike Niehs, Arne Schmidt, Christian Scheffer, Daniel E. Biediger, Michael Yannuzzi, Benjamin Jenett, Amira Abdel-Rahman, Kenneth C. Cheung, Aaron T. Becker, and Sándor P. Fekete. Recognition and reconfiguration of lattice-based cellular structures by simple robots. In *International Conference on Robotics and Automation (ICRA)*, pages 8252–8259, 2020. doi:10.1109/ICRA40945.2020.9196700.

[39] Ana L. Santos, Dongdong Liu, Anna K. Reed, Aaron M. Wyderka, Alexis van Venrooy, John T. Li, Victor D. Li, Mikita Misiura, Olga Samoylova, Jacob L. Beckham, Ciceron Ayala-Orozco, Anatoly B. Kolomeisky, Lawrence B. Alemany, Antonio Oliver, George P. Tegos, and James M. Tour. Light-activated molecular machines are fast-acting broad-spectrum antibacterials that target the membrane. *Science Advances*, 8(22):eabm2055, 2022. doi:10.1126/sciadv.abm2055.

[40] Zhuoran Zhang, Xian Wang, Jun Liu, Changsheng Dai, and Yu Sun. Robotic micromanipulation: Fundamentals and applications. *Annual Review of Control, Robotics, and Autonomous Systems*, 2:181–203, 2019. doi:10.1146/ANNUREV-CONTROL-053018-023755.

[41] Yangsheng Zhu, Mingjing Qi, Zhiwei Liu, Jianmei Huang, Dawei Huang, Xiaojun Yan, and Liwei Lin. A 5-mm untethered crawling robot via self-excited electrostatic vibration. *IEEE Transactions on Robotics*, 38(2):719–730, 2022. doi:10.1109/TRO.2021.3088053.