Regular Paper

Hardness of Reconfiguring Robot Swarms with Uniform External Control in Limited Directions

David Caballero[1]   Angel A. Cantu[1]   Timothy Gomez[1]   Austin Luchsinger[1,a]
Robert Schweller[1]   Tim Wylie[1]

Abstract: Motivated by advances in microscale applications and simplistic robot agents, we look at problems based on using a global signal to move all agents when given a limited number of directional signals and immovable geometry. We study a model where unit square particles move within a 2D grid based on uniform external forces. Movement is based on a sequence of uniform commands which cause all particles to move 1 step in a specific direction. The 2D grid board additionally contains "blocked" spaces which prevent particles from entry. Within this model, we investigate the complexity of deciding 1) whether a target location on the board can be occupied (by any) particle (*Occupancy problem*), 2) whether a specific particle can be relocated to another specific position in the board (*Relocation problem*), and 3) whether a board configuration can be transformed into another configuration (*Reconfiguration problem*). We prove that while occupancy is solvable in polynomial time, the relocation and reconfiguration problems are both NP-Complete even when restricted to only 2 or 3 movement directions. We further define a hierarchy of board geometries and show that this hardness holds for even very restricted classes of board geometry.

Keywords: robot motion planning, computational complexity, block puzzle

1. Introduction

The tilt model, proposed by Becker et al. [5], has foundations in classical robot motion planning. This model consists of a 2D grid of open and blocked spaces, called the "board", along with a set of unit square pieces/tiles placed at open board locations. A sequence of "tilts" pushes all the board pieces maximally in a specified cardinal direction. A sequence of such tilts transforms the initial board configuration into a new configuration. Some natural computational problems related to this model are those of *occupancy*, *relocation*, and *reconfiguration*. *Occupancy* is the problem of determining if there exists a sequence of tilts such that a specific, initially empty, board location may be occupied by a particle on the board. *Relocation* is the problem of whether a sequence of tilts exists to relocate a specific tile from location *a* to location *b*. *Reconfiguration* asks if a sequence of tilts exists to transform board configuration *A* to board configuration *B* (where each configuration specifies the location of all tiles on the board). These problems were recently all shown to be PSPACE-Complete (in 4-directions) [2].

Here, we discuss a variant of this model (introduced in Ref. [4]) in which particles only move 1 step per tilt in the specified direction, rather than maximally. **Figure 1** shows a simple example. We further consider these problems with limited usable directions (e.g., only tilting down and right), as well as considering the effect of limiting the complexity of the geometry of the open spaces

of the board. For example, one limited type of geometry is that in which the open spaces form an "x/y-monotone" shape.

1.1 Previous Work

The problems of Occupancy, Relocation, and Reconfiguration were first studied in Ref. [5] in the full tilt model. In this work, the authors showed NP-hardness for the Occupancy Problem. Soon after, the authors of Ref. [7] showed that finding the minimum move sequence for reconfiguring one configuration to another is PSPACE-Complete. Additional algorithmic, complexity, and logic work was done in Ref. [6]. All of these results used only 1×1 pieces. Later work in Ref. [1] relaxed the constraint on tile size and showed the Relocation and Reconfiguration Problems were PSPACE-Complete when only a single 2×2 polyomino is allowed. Recent work strengthened these results and showed PSPACE-Completeness for all three problems even when only allowing 1×1 pieces [2]. Additional work has focused on the application of the tilt model for the assembly of general shapes [8], [10], [11], including *universal constructors* [1], [2], and sorting polyominoes [9].

1.2 Our Contributions

We investigate some natural questions related to these problems and seek to find simple versions that are still computationally intractable. We remove the requirement for tiles to slide maximally and focus on unit movements as in Ref. [4]. In this model, the occupancy problem is solvable in polynomial time

---
[1]   The University of Texas Rio Grande Valley, Edinburg, Texas 78539, USA
[a]   amluchsinger@gmail.com

Table 1 An overview of results. The items in column **Problem** list the problems explored in this model. **Direct.** indicates how many directions of movement are allowed for the result. For the case of 2 directions we consider orthogonal directions. **Geometry** describes the type of board geometry used for the reduction (or allowed by the algorithm). The **Result** column shows the complexity of the problem with the theorem being found in column **Theorem**.

Problem	Direct.	Geometry	Result	Theorem
Occupancy	Any	All	P	Thm. 3.1
Relocation	2, 3	Monotone	NP-Complete	Thm. 5.1
Reconfiguration	2	Connected	NP-Complete	Thm. 6.1

(Theorem 3.1), so we focus on the Relocation and Reconfiguration problems. We show intractability based on restricted directions for both reconfiguration and relocation. We show that the Relocation problem is NP-Complete even when limited to monotone board geometry (Theorem 5.1) and the Reconfiguration Problem is NP-Complete when limited to Connected boards (Theorem 6.1). A summary of our results is shown in **Table 1**.

2. Preliminaries

Board. A *board* (or *workspace*) is a rectangular region of the 2D square lattice in which specific locations are marked as *blocked*. Formally, an $m \times n$ board is a partition $B = (O, W)$ of $\{(x, y) | x \in \{1, 2, \ldots, m\}, y \in \{1, 2, \ldots, n\}\}$ where O denotes a set of *open* locations, and W denotes a set of *blocked* locations- referred to as "concrete." We classify the different board geometries according to the following hierarchy:

- Connected: A board where the set of open spaces O is a connected shape.
- Simple: A connected board is *simple* if O has genus-0, meaning that the set of open spaces is a shape without holes.
- Monotone: A simple board where O is either horizontally or vertically monotone.
- Convex: A monotone board where O is both horizontally and vertically monotone.
- Rectangular: A convex board where O is a rectangle.

Configurations. A configuration is an arrangement of tiles placed on open locations of a given board. Formally, a *tile* is a labeled unit square centered on a non-blocked point on a given board. A *configuration* $C = (B, P = \{P_1 \ldots P_k\})$ consists of a board B and a set of non-overlapping tiles P that each do not overlap with the blocked locations of board B.

Step. A *step* is a way to turn one configuration into another by way of a global signal that moves all tiles in a configuration one unit in a direction $d \in \{N, E, S, W\}$ when possible without causing an overlap with a blocked position or another tile. Formally, for a configuration $C = (B, P)$, consider the translation of all tiles in P by 1 unit in direction d. If no overlap with blocked board spaces occurs, then the new configuration is derived by first performing this translation. On the other hand, for each tile for which the translation causes an overlap with a blocked space, temporarily add these tiles to the set of blocked spaces and repeat. Once the translation induces no overlap with blocked spaces, execute the translation of the remaining non-blocked tiles to arrive at the new configuration. If all tiles are eventually marked as blocked spaces, then the step transition does not change the initial configuration.

We say that a configuration C can be *directly reconfigured* into
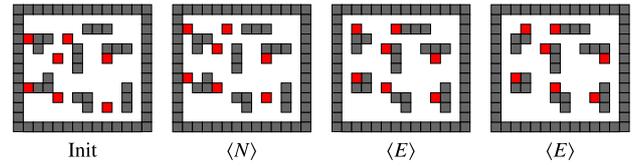
Fig. 1 An example step sequence. The initial board configuration followed by the resulting configurations after an N step, E step, and then final E step. So, the sequence to get from the initial state to the final state in this figure is $\langle N, 2E \rangle$.

configuration C' (denoted $C \rightarrow_1 C'$) if applying one step in some direction $d \in \{N, E, S, W\}$ to C results in C'. We define the relation $\rightarrow_*$ to be the transitive closure of $\rightarrow_1$ and say that C can be *reconfigured* into C' if and only if $C \rightarrow_* C'$, i.e., C may be reconfigured into C' by way of a sequence of step transformations.

Step Sequence. A *step sequence* is a sequence of steps which can be inferred from a sequence of directions $D = \langle d_1, d_2, \ldots, d_k \rangle$; each $d_i \in D$ implies a step in that direction. For simplicity, we often just refer to the sequence of directions from which that sequence was derived. Given a starting configuration, a step sequence corresponds to a sequence of configurations based on the step transformation. *Note:* As a shorthand, when sequential steps $d_i, d_{i+1}, \ldots, d_{i+n}$ in a sequence are all in the same direction, we simply annotate those steps with $n * d$ (e.g. $\langle N, N, N \rangle$ is written as $\langle 3N \rangle$). An example is shown in Fig. 1.

2.1 Problem Definitions

In this paper we consider the following three problems under the step transformation. In subsequent theorem statements we will state the complexity of these problems "under the step transformation" to help compare these results with work done "under the tilt transformation".

Occupancy. The occupancy problem asks whether or not a given location can be occupied by any tile on the board. Formally, given a configuration $C = (B, P)$ and a coordinate $e \in B$, does there exist a C' such that $C \rightarrow_* C'$ where $C' = (B, P')$ and $\exists p \in P'$ that contains a tile with coordinate e?

Relocation. The relocation problem asks whether a specified tile can be relocated to a particular position. That is, given a configuration, a specific tile within that configuration, and a translation of that tile, does there exist a sequence of steps which moves the original tile to its translation?

Reconfiguration. The reconfiguration problem asks whether a configuration can be reconfigured into another. Formally, given two configurations $C = (B, P)$ and $C' = (B, P')$, does $C \rightarrow_* C'$?.

3. Occupancy

Theorem 3.1. *The occupancy problem under the step transformation is in P when using only 1×1 tiles regardless of direction limitations. In particular, the problem is solvable in $O(|B|)$ time, where $|B|$ denotes the number of positions on the input board B.*

Proof. Perform a breadth-first-search from the goal location to determine if any position containing a tile is reachable by way of a sequence of unit distance steps between north/south or east/west connected open locations. If no tiled position is reachable, the goal position is clearly not occupiable. If there is a reachable tile, consider the closest such tile. The shortest path connecting this

tile to the goal location yields a tilt sequence that is guaranteed to place the tile at the goal, as the only way for it to be blocked would require another tile to move into position ahead of it along the shortest path, contradicting the claim that the first tile was the closest to the goal position. Thus, the occupancy problem can be solved in $O(|B|)$ time.   □

4. NP-Membership with Limited Directions

In this section we prove NP membership for the relocation and reconfiguration problems when restricting steps to two orthogonal directions. The directions we consider, south and east, can be defined as the set $d = \{S, E\}$. Therefore, tiles can only move to a position that is south, east or southeast of the starting spot. We consider these directions since the case when the directions are opposite each other is solvable in polynomial time. For three directions we use west, south, and east, which can be represented by the set $d = \{W, S, E\}$. The following will consider a configuration with n tiles and a board of size $l \times w$.

Membership comes from the polynomial upper-bound on the length of a move sequence using limited directions.

Lemma 4.1. For any two configurations C and C', if $C \rightarrow_* C'$ through a sequence of south and east steps, there exists a step sequence S of length $O(n(l + w))$ using only those two directions such that applying S to C results in C'.

Proof. Consider a tile t at location p in C and at p' in C', the maximum distance between p and p' is $l+w$. Any step that moves t brings it closer to p' since p' must be to the south east of its starting location p, if a step moves t further away from p' than that step is not in S since it would have moved t to a location such that it can never reach p'. Now notice that each step must move at least one tile, and any step that moves a tile must move it closer to its position in C'. Since each tile can move at most $l + w$ and each step must move at least one tile, if C can be reconfigured to C' then it will be able to do so in $O(n(l + w))$ steps.   □

Lemma 4.2. For any two configurations C and C', if C can be reconfigured to C' using only west, south, and east steps, there exists a step sequence S of length $O(nlw^2)$ using only those three directions such that applying S to C results in C'.

Proof. First let us consider the maximum number of south movements that can be made. Similar to above each south step moves at least one tile further south so the maximum number of south steps that can be made is nl. Now considering the directions east and west, the maximum number of consecutive steps that can be made in one direction is w. Any step can be undone by making a step in the opposite direction unless the first step changed the position of tiles relative to each other. However this only brings the maximum move sequence for east and west to be w^2. Since the maximum move sequence before having to make a south step is w^2 and the maximum number of south steps is nl the maximum move sequence length is $O(nlw^2)$.   □

Theorem 4.3. When restricted to 2 or 3 directions, the Relocation and Reconfiguration problems under the step transformation are in NP.

Proof. A step sequence can be used to verify positive answers to the Relocation and Reconfiguration problems. From Lemmas 4.1 and 4.2 we can see that the step sequences for the cases of 2 and

3 directions are polynomially bounded.   □

5. Relocation with Limited Directions

In this section we detail a reduction from 3SAT to relocation in monotone boards using limited directions. We refer to the position the tile destined for relocation is initialized as location a and the goal position as location b. W.l.o.g, the directions used in the reduction are limited to *east* and *south*.

Overview. The board of this reduction consists of a long hallway-like structure with notches that can fit some tiles and prevent them from moving. The tile we are attempting to relocate begins at the top left position of the board with its goal location being at the bottom right. In between both locations is a section of the board that maps to the clauses of a 3SAT instance, containing tiles that are mapped to the literals of each clause. These tiles mapped to literals block the path from the tile we are attempting to relocate to the goal location at the bottom right part of the board. Certain step sequences are performed so that truth assignments are made to the literals, causing the tiles mapped to the literals of the clause chains to become stuck in some notches if they are set to true. If the proper variable assignments are made, then the path will be cleared, and the tile we are relocating will be able to arrive at the goal location.

5.1 Relocation Preliminaries

We first describe the gadgets used for this reduction, as well as present some notation that we use throughout the section.

Gadgets. All of the gadgets discussed in this section are shown in **Fig. 2**. We will later describe how to construct a board by attaching gadgets together via their north (head) and south (foot) *binding locations* (depicted in blue). The relocation tile is located at position a inside of the start gadget. The Assignment gadget is where tilts will correspond to variable assignments for our reduction. Every literal of a 3SAT formula has a corresponding Positive or Negative gadget, along with an associated tile that represents that literal. Location b resides inside the Goal gadget, a gadget whose structure grows with respect to the number of variables N. The area within the Goal gadget that grows based on N is outlined in Fig. 2. The Buffer and Notch gadget are utility gadgets discussed when needed.

Notation. Here we present the notation for describing the construction of a board with the gadgets described above. We define $g_i \searrow g_j$ as gadget g_i's foot binding with gadget g_j's head. A *chain* is defined as a sequence of gadgets $S = \langle g_1, g_2, \ldots g_n \rangle$, s.t.
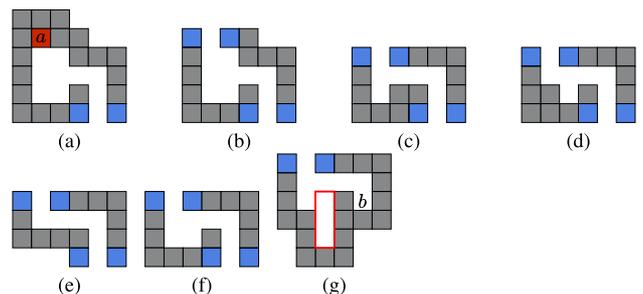
Fig. 2 (a) Start gadget (b) Assignment gadget (c) Positive gadget (d) Negative gadget (e) Buffer gadget (f) Notch gadget (g) Goal gadget.

$g_1 \searrow g_2, g_2 \searrow g_3, \ldots, g_{n-1} \searrow g_n$. It follows that a chain also has binding locations that allow it to bond with other chains or gadgets. We define $G = \langle S_1, S_2, \ldots, S_n \rangle$ as a *sequence of chains* s.t. $S_1 \searrow S_2, S_2 \searrow S_3, \ldots, S_{n-1} \searrow S_n$. Similarly, we say the binding of two sequences of chains $G = \langle S_1, S_2, \ldots, S_n \rangle$ and $G' = \langle S'_1, S'_2, \ldots, S'_m \rangle$ yields another sequence of chains $G'' = \langle S_1, S_2, \ldots, S'_{m-1}, S'_m \rangle$ s.t $S_1 \searrow S_2, \ldots, S_n \searrow S'_1, \ldots, S'_{m-1} \searrow S'_m$.

5.2 Board

The board consists of three sections: the *assignment*, *formula*, and *validation* section. The tile initialized at location a within the assignment section is referred to as the *relocation* tile, the tiles inside the formula section are referred to as the *literal* tiles, and the tiles within the validation section are called the *validation* tiles. These board sections are all chains of the gadgets shown in Fig. 2.

Validation Section. The validation section is the sequence of gadgets $S_V = \langle g_1, g_2, \ldots, g_N \rangle$ where:
- $g_N = Goal$ gadget and $\forall i < N$, $g_i = Buffer$ gadget with a validation tile allocated in-between the head of every gadget.

Formula Section. For the set of clauses C and set of variables $X = \{x_1, x_2, \ldots, x_N\}$, the clause chain of clause $c \in C$ is the sequence $G_c = \langle S_1, S_2, S_3, S_4, S_5 \rangle$ where:
- $S_1 = \langle g_1, g_2, \ldots, g_N \rangle$ where if the literal $l_p \in c$ is the variable $x_p \in X$, then $g_p = Positive$ gadget if l_p is a positive literal or $g_p = Negative$ gadget if l_p is a negative literal. The literal tile for literal l_p is allocated in between the head of g_1.
- $S_2 = \langle g_1, g_2, \ldots, g_N \rangle$ where if the literal $l_q \in c$ is the variable $x_q \in X$, then $g_q = Positive$ gadget if l_q is a positive literal or $g_q = Negative$ gadget if l_q is a negative literal. The literal tile for literal l_q is allocated in between the head of g_1.
- $S_3 = \langle g_1, g_2, \ldots, g_N \rangle$ where if the literal $l_r \in c$ is the variable $x_r \in X$, then $g_r = Positive$ gadget if l_r is a positive literal or $g_r = Negative$ gadget if l_r is a negative literal. The literal tile for literal l_r is allocated in between the head of g_1.
- $S_4 = \langle g_1, g_2, \ldots, g_{2N+1} \rangle$ where $\forall i < 2N+1$, $g_i = Buffer$ gadget, and $g_{2N+1} = Notch$ gadget.
- $S_5 = \langle g_1, g_2, \ldots, g_{2N+1} \rangle$ where $\forall i$, $g_i = Buffer$ gadget.

The formula section is therefore the sequence $G_F = \langle G_{c_1}, G_{c_2}, \ldots, G_{c_{|C|}} \rangle$ where every $G_{c_i} \in G_F$ is a clause chain and $G_{c_1} \searrow G_{c_2}, G_{c_2} \searrow G_{c_3}, \ldots, G_{c_{|C|-1}} \searrow G_{c_{|C|}}$

Assignment Section. The assignment section is a sequence of chains $G_A = \langle S_1, S_2, S_3, S_4 \rangle$ where:
- $S_1 = \langle g_1, g_2, \ldots, g_N \rangle$ where $g_1 = Start$ gadget and $\forall i > 1$, $g_i = Assignment$ gadget. The relocation tile is initialized within the Start gadget, as depicted in Fig. 2.
- $S_2 = \langle g_1, g_2, \ldots, g_{2N} \rangle$ where $\forall i$, $g_i = Notch$ gadget.
- $S_3 = \langle g_1, g_2, \ldots, g_{2N+1} \rangle$ where $g_1, g_{N+1}, g_{2N+1} = Assignment$ gadget and $\forall i$ s.t $g_i \neq Assignment$ gadget, $g_i = Notch$ gadget.
- $S_4 = \langle g_1, g_2, \ldots, g_v \rangle$ where $v = |S_V| + |G_F|$, and $\forall i \leq v$, $g_i = Notch$ gadget.

The board $B = \langle G_A, G_F, S_V \rangle$ is therefore the combination of the three sections where $G_A \searrow G_F$ and $G_F \searrow S_V$.

(a) $\langle 2S, 3E, 2S \rangle$

(b) $\langle S, E, S, 2E, 2S \rangle$
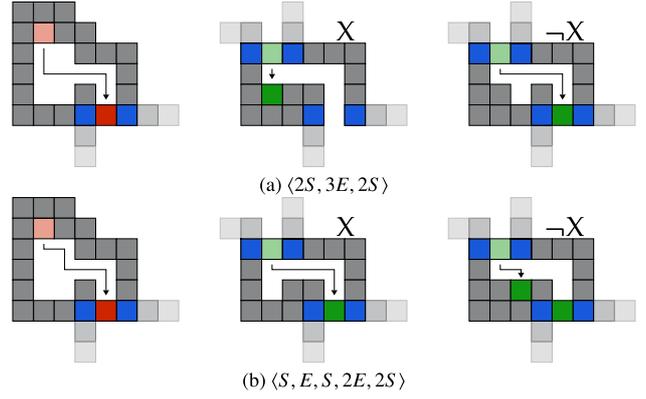
Fig. 3 Assigning (a) true or (b) false to some variable x. Setting a literal to true is done by confining its corresponding literal tile within a Positive or Negative gadget.

5.3 Reduction

The reduction can be understood as a two phase process. In the first phase variables are given a truth value one-by-one in ascending order for the set of variables $X = \{x_1, x_2, \ldots, x_N\}$. The second phase consists of verifying if the variable assignments from the first phase satisfied the clauses of the formula. Every clause chain is checked for any unconfined literal tiles, where if a clause chain confines at least one literal tile it is said to be satisfied, and unsatisfied otherwise. A satisfied clause chain can have up to two remaining unconfined literal tiles, where an unsatisfied clause will have three unconfined literal tiles. In order to identify satisfied and unsatisfied clauses, the remaining literal tiles are 'counted' by attempting to occupy two open spaces using the literal tiles in each of their respective clause chains. Attempting to occupy the two spaces with three literal tiles (e.g., an unsatisfied clause chain) will leave one literal tile unconfined which will be used to prevent relocation.

First Phase. Starting with the board configuration described above, each variable in ascending order can be assigned some truth value by executing the step sequences depicted in **Fig. 3**. During this phase, the relocation tile will visit every gadget inside the first chain of the assignment section if either one of the sequences depicted is executed for every variable truth assignment.

To ensure the step sequences depicted are the only ones executable during this phase, the validation tiles in the validation section will enter the Goal gadget one-by-one for every assignment and risk occupying location b, as shown in **Fig. 4**. This relation between the relocation and validation tiles ensures that the two sequences depicted are the only ones performed during this phase.

All literal tiles visit every gadget of the chain it resides in for every clause chain (one of which is a Positive or Negative gadget) as the step sequence depicted above is performed. By design, the literal tiles reach a Positive or Negative gadget when its corresponding variable is next in the truth assignment procedure (**Fig. 5**). After the last variable is assigned some truth value, every clause chain will confine at least one literal tile if the corresponding clause was satisfied, and zero literal tiles if it was not.

Second Phase. Immediately after the first phase the relocation tile will reside inside the second chain of the assignment section,
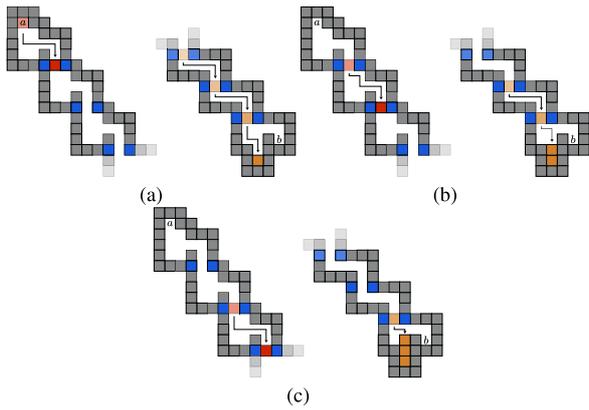
Fig. 4 Illustration of validation tiles entering the Goal gadget as truth assignments are made. In (a) and (c), true assignments are made, showing the start and end location of the tiles. In (b), a false assignment was made, showing how the validation tiles are still placed outside of location b.

(a) $x_1 \to 1$

(b) $x_2 \to 0$

(c) $x_3 \to 1$
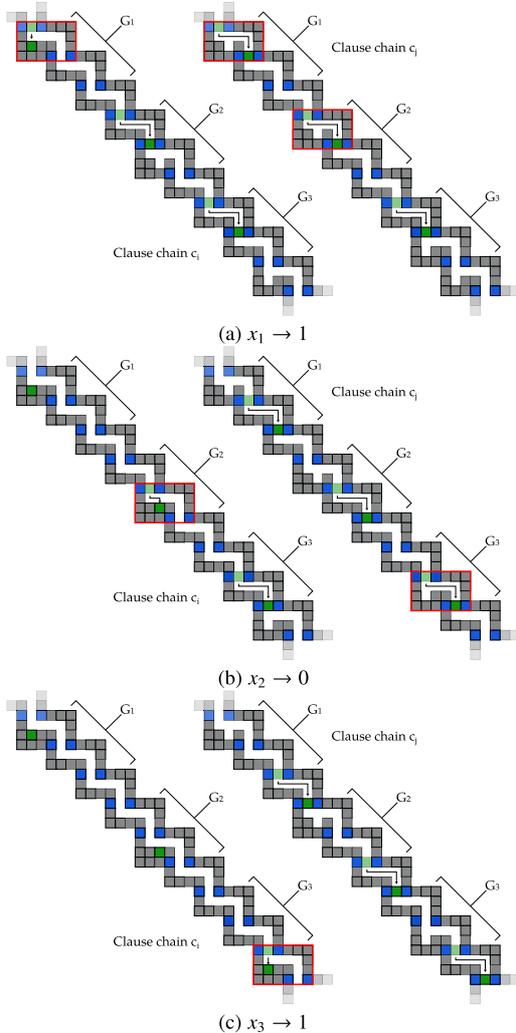
Fig. 5 Visualization of two clause chains $c_i = (x_1 \vee \neg x_2 \vee x_3)$, $c_j = (\neg x_1 \vee \neg x_1 \vee x_2)$ for $X = \{x_1, x_2, x_3\}$ during the first phase. (a) Assigning the variable x_1 to true value is done by the step sequence $\langle S, 3E, 2S \rangle$, which makes all other tiles in the clauses chains and assignment section move to the next gadget. (b) Assigning the value *false* to the variable x_2 is done by the sequence $\langle E, S, 3E, 2S \rangle$. (c) By performing the true sequence again, the variable x_3 can be assigned some truth value as well.

(a)

(b)

(c)
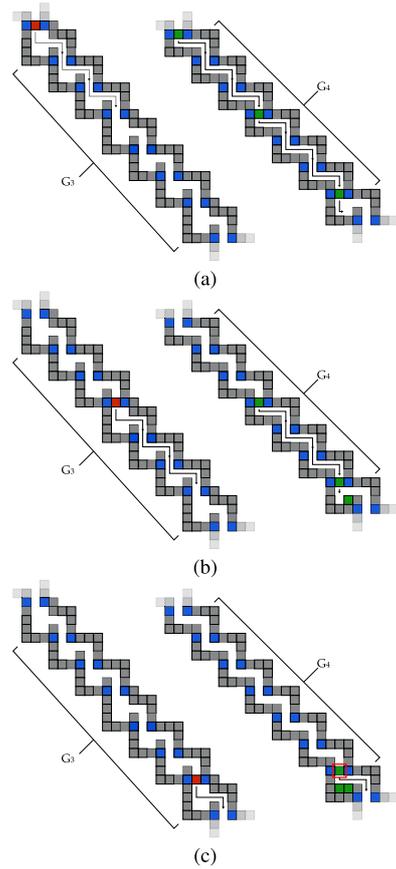
Fig. 6 During the second phase, unconfined literal tiles are forced inside the Notch gadget of the fourth chain of their corresponding clause chain. Unsatisfied clause chains export one too many literal tiles, as shown in (c).

whose structure consists of only Notch gadgets. The Notch gadget serves two functions in this phase, one of which is to forbid the movement in the *south* direction when the relocation tile traverses through it. The first step in this phase is therefore to execute the step sequence $\langle S, 3E, 2S \rangle$ until the relocation tile reaches the third chain, which simultaneously moves any unconfined literal tiles to the fourth chain of their respective clause chains.

Afterwards, the relocation and any unconfined literal tiles will reside inside the third and fourth chain, respectively, of their corresponding section/chain, as depicted in **Fig. 6**. The last gadget of the fourth chain of every clause chain consists of a Notch gadget, whose function in this case is to provide two open spaces for the unconfined literal tiles from the clause chains. Moving the relocation tile through the third chain of the assignment section, each remaining literal tile will reach the last Notch gadget of their corresponding clause chain and fit in one of the two open spaces made by the Notch gadget, provided that the clause chain was satisfied. If a clause chain was unsatisfied, then the third literal like will not fit within the two open spaces provided by the Notch gadget, leaving it free to continue through the board. At this point, the relocation tile would have reached the fourth chain of the assignment section, which is composed entirely of Notch gadgets and whose length is equal to that of the length of the formula section plus the length of the validation section. Traversing the tile through the fourth chain will move any free literal tiles towards the validation section, blocking location b. Therefore, if there

exists at least one unsatisfied clause chain, there exists at least one unconfined literal tile inexorably pushed to occupy location b whilst the relocation tile traverses through the architecture of the fourth chain of the assignment section. If there are no unsatisfied clause chains, it follows that there will be a direct path from the relocation tile to location b on the board allowing its relocation there.

5.4 Theorem

Theorem 5.1. The relocation problem under the step transformation is NP-Complete on a monotone board when limited to two directions.

Proof. Membership in NP is described in Section 4. To show NP-Completeness, a reduction from 3SAT is detailed to show relocation in monotone boards solves 3SAT. For a given 3SAT instance, a monotone board $B = (O, W)$ is designed such that the elements of the 3SAT instance are represented by corresponding elements of the monotone board. The elements of a given 3SAT instance consist of the set of clauses along with the literals that make up each individual clause. The literals of each clause are represented by individual tiles, which inhabit their pertaining clause chain. As stated above, the literals can be assigned some truth value by either confining the associated tile within its pertaining clause chain (e.g., assigning true) or keeping it unconfined (e.g., assigning false). The reduction can be understood as a two phase process:

Phase one. Starting with an initial board $B = (O, W)$ described in the previous section, the first phase consists of making variable truth assignments in ascending order for the set of Boolean variables $X = \{x_1, x_2, \ldots, x_N\}$. Each variable can be given some truth value by performing either step sequence illustrated in Fig. 3. Simultaneously, each individual literal tile in the clause chains will reach a corresponding Positive or Negative gadget exactly when the variable it equals is assigned some truth value. When assigning a variable to true, the positive literals that equal that variable will become confined, whereas the negative literals that equal that variable will remain unconfined. Likewise, assigning a variable to false keeps the positive literals unconfined and the negative ones confined. During this phase, the design of the validation section along with the inhabitant validation tiles restrict the available step sequences to that of the two previously mentioned. Given that the Goal gadget resides within the validation section, along with location b of the board, the validation tiles risk occupying location b unless either of the step sequences are performed during this phase. The restrictive nature of the validation section ensures that phase one of the reduction consists only of valid variable truth assignments.

Phase two. Afterwards, every clause chain assumes some satisfiability status by the quantity of confined literal tiles residing within them. During this phase, the number of unconfined literal tiles of each clause chain are 'counted' by the attempt to occupy two open spaces inside their corresponding clause chains with these literal tiles. A satisfied clause chain will confine at least one literal tile and leave at most two literal tiles unconfined. These two unconfined literal tiles are able to occupy the two open spaces, essentially removing them from the path to location b

from location a. On the other hand, an unsatisfied clause yields one too many unconfined literal tiles for the two corresponding open spaces, causing one of the literal tiles to remain within the path to location b from location a.

Following this two phase process, the tile initialized at location a is relocatable to location b on the board if every clause chain yielded at most two unconfined literal tiles after the first phase. During the second phase, these two unconfined literal tiles can be removed from the path from location a to location b, allowing access for the relocation tile to the targeted location on the board. Similarly, relocation becomes impossible when a clause chain produces three unconfined literal tiles after phase one since the attempt to relocate after the second phase will inexorably occupy location b with one of the unconfined literal tiles. It follows that the relocation of the relocation tile initialized at location a is possible if all clause chains were satisfied, which is possible only with a satisfying truth assignment to the set of variables. Moreover, if the boolean formula is not satisfiable, then some clause chains will always yield three unconfined literal tiles. The forward-progressing nature of the construction, along with the inability to store tiles in improper notches, means that the relocation tile can never be relocated to its goal position (because it is blocked). Therefore, for a given 3SAT instance relocation is possible if and only if the boolean formula is satisfiable. □

Theorem 5.2. The relocation problem under the step transformation is NP-Complete on a monotone board when limited to three directions.

Proof. If we allow for movement in a third direction (e.g., movement in the $\langle W \rangle$ direction), we see how the phases previously described are not significantly changed. During phase one, the validation section is used to restrict the step sequences available when assigning truth values to variables where the inclusion of the third direction only adds the cycling between the choices of the step sequences (assigning true or assigning false). Therefore making movement in the third direction available does not change the way variable truth assignments are made. Moreover, the manner in which tiles traverse through the board remains the same since the movement in the third direction will simply revert the board back to the configuration that was previously visited when movement in the $\langle E \rangle$ direction was performed. Thus, the phases of the reduction persist even with the third direction. □

6. Reconfiguration with Limited Directions

We now will take a look at the problem of reconfiguration on a connected board when limited to two orthogonal directions. We show that with these constraints the problem is NP-Complete under the step transformation. Without loss of generality we will be limiting the directions to south and east. In this section, we prove hardness with a reduction from 3SAT. The following subsections describe the gadgets used when constructing a tilt configuration from a 3SAT formula with n variables and m clauses. We present individual gadgets separately which can be connected in a way that does not effect their functionality.

6.1 Gadget Construction

In this subsection, we describe how to construct an instance of
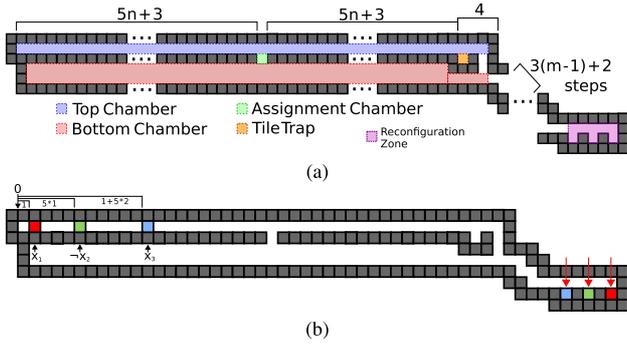
Fig. 7 (a)3 SAT clause gadget, where n is the total number of unique variables, and m indicates that this clause is the mth clause in the formula. (b) Example of variable placement for clause $(x_1 \lor \neg x_2 \lor x_3)$ in a 5 variable formula. Goal locations indicated by red arrows.
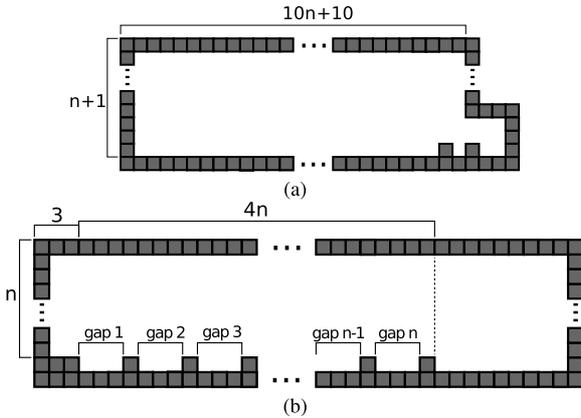
Fig. 8 (a) South Limiter: Limits the amount of south steps made before all variables have been assigned. (b) South Forcer: Forces the user to make south steps at specific times.

the reconfiguration problem from an instance of the 3SAT problem. We discuss this construction in terms of gadgets.

Clause Gadget Each clause of the 3SAT formula will be represented by a clause gadget (depicted in **Fig. 7**). This gadget consists of a top chamber in which variable tiles will be placed and a 1-wide gap that connects that chamber to a second bottom chamber. There will be a staircase that connects the lower chamber to a final reconfiguration zone, which contains the goal locations for the three variable tiles in the clause. The number of steps of the staircase will be a function of the current clause; so, the mth clause in the formula will have a staircase with $3(m-1)+2$ steps.

For every variable x_i, in every clause that it appears, we will place a corresponding variable tile in the top chamber of its respective clause gadget at position $5(i-1)+1$ for every positive variable and $5(i-1)$ for every negated variable. The goal locations of the tiles will be one in each of the notches. Without loss of generality, the goal locations will be in the reverse order from west to east that they start at in the top chamber. Figure 7 shows an example tile placement for a clause gadget.

South Limiter Gadget Every instance of the reconfiguration problem obtained through this reduction will contain a single *south limiter gadget* (shown in **Fig. 8**). The purpose of this gadget is to limit the number of *south* steps that can be made in the reconfiguration step sequence. The height and width of this gadget will scale linearly to the number of variables in the 3SAT formula.

In the South Limiter gadget we place a tile at the northwest
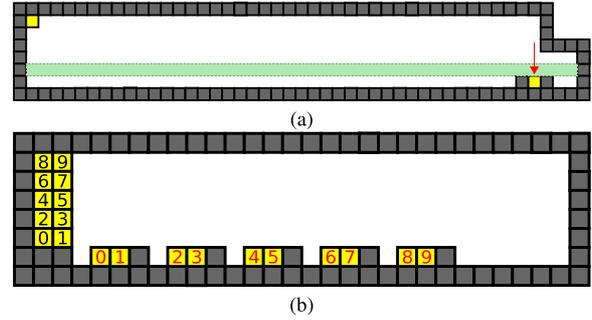
Fig. 9 (a) Example of south limiter tile placement in a 5 variable formula. Goal location indicated by red arrow. Post assignment zone highlighted in green. (b) Example of south forcer tile placement in a 5 variable formula. Goal location indicated by red labels.

corner, and the goal location is the notch at the other end of the gadget. **Figure 9** depicts tile and goal placement for this gadget.

South Forcer Gadget Each instance will also have one *south forcer gadget* (shown in Fig. 8). The purpose of this gadget is to enforce the proper timing of any *south* step made in the reconfiguration step sequence. The height and width of this gadget also scales linearly to the number of variables.

In the South Forcer gadget we place 2 tiles in a row for every distinct variable in the 3SAT formula on the west side of the gadget. The goal location of row n is the eastmost side of the nth gap 8, in the same position relative to each other that they started in. Figure 9 depicts tile and goal placement for this gadget.

6.2 Gadget Functionality

Here, we discuss the relationship between our tilt reconfiguration and the 3SAT variable assignment, as well as the functionality of each of the gadgets presented in the previous subsection.

Variable Assignment This reduction works by utilizing the uniform global signals to assign all variables x_i a truth value simultaneously. To start the process of reconfiguration we will begin stepping east, uniformly shifting all variable tiles along the top chamber of the clause gadgets. When the eastmost variable tile is located above the assignment chamber, the choice of assigning that variable true can be made by performing step sequence $s_t = \langle S, E \rangle$, or false with the step sequence $s_f = \langle E, S \rangle$. A variable tile evaluates to true if it enters the assignment chamber. Since all variable tiles t_i representing a particular variable x_i share the same x-coordinate, all of these tiles will receive the same assignment.

South Forcer/Limiter Gadgets The reconfiguration requirement of the *south forcer gadget* ensures that every variable must receive an assignment. The only way to place each tile-pair in their respective goal positions is to "assign" each of the n variables a value of either true or false. The *south limiter gadget* ensures that only n assignments can be made, since doing more would position the gadget's tile along the bottom edge of the gadget with no way of reaching its goal position. The combination of these gadgets prevents the possibility of assigning both a positive and negated variable tile, i.e. t_i and $\neg t_i$, values of true. Once every variable tile has been assigned, the *south limiter gadget's* tile will be in the *post assignment zone* (see Fig. 9). In order to achieve the configuration goal inside that gadget, only east steps can be
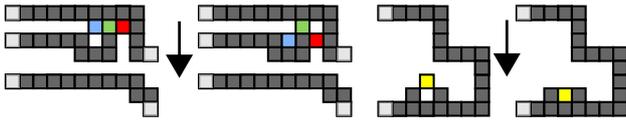
Fig. 10 State of a clause gadget in which no variable tiles are evaluated to a literal true before and after the forced south tilt, and the respective states of the *south limiter gadget*.

made until the tile is directly above its goal location. With global signals this causes all variables tiles to be pushed maximally to the east side of the top chamber of their respective clause gadget. Once the *south limiter's* tile is above its goal location, a south step must be performed, since we are limited to only south and east, and an east step would make the south limiter's tile pass its goal position.

Clause Verification A clause gadget is in its goal configuration when each of its variable tiles are in their goal locations at the bottom right of the gadget. A clause gadget will be unable to reach its goal configuration if no variable tiles make it through the assignment chamber (this is the equivalent to every literal of a 3SAT clause evaluating to false). If there are three variable tiles still in the top chamber of a clause gadget, a south step will place the westmost variable tile into the *tile trap*. This leaves the board unreconfigurable (see **Fig. 10**). If at least 1 variable tile is passed into the assignment chamber, then the clause gadget will always be reconfigurable. From there the tiles can be moved to their respective staircase, where the varied number of steps allows for each tile to be moved to its goal location individually.

6.3 Formal Proof

Theorem 6.1. The reconfiguration problem under the step transformation is NP-Complete on a connected board when limited to two directions.

Proof. To show hardness for the reconfiguration problem under the step transformation, we reduce from 3SAT. Given a 3SAT formula consisting of n variables and m clauses, we construct a tilt configuration using the gadgets described in Section 6.1. We create a clause gadget for every clause in the 3SAT formula, as well as a single South Forcer Gadget and a single South Limiter Gadget. To make this a connected board these gadgets can be attached at their northwest corner without affecting their functionality.

First, we argue that the instance of the reconfiguration problem obtained through this reduction is solvable if the respective 3SAT formula is solvable. This follows from the variable assignment paragraph in Section 6.2. Given a certificate containing the truth values that satisfy the 3SAT formula, start the reconfiguration by stepping east. When variable tile t_i is up for assignment, i.e., the positive variable tile representing x_i is above the assignment chamber, a $\langle S, E \rangle$ step sequence can be input if the variable x_i is being set true, and a $\langle E, S \rangle$ step sequence if the variable x_i is to be assigned false. Since the certificate contains a valid solution to the 3SAT formula, then at least one tile in every clause will evaluate to true (following the functionality section), meaning all clause tiles can be placed in their goal locations. Furthermore, the tiles in the south limiter gadget and the south forcer gadget will reach their goal positions via the standard variable assignment.

Next, we argue that the respective 3SAT problem is solvable if

the obtained instance of the reconfiguration problem has a solution. This direction relies on the functionality of the South Limiter and South Forcer gadgets, as well as the clause verification discussed in Section 6.2. We define d_a as the distance between the westmost variable tile (x_1 or $\neg x_1$ if it exists) and the assignment chamber. We observe the distance from the tile in the *south limiter gadget* to its goal location is greater than d_a. Only n south steps can be input before the south limiter gadget's tile is above its goal location. Every variable tile assignment requires a south step, and once the *south limiters gadget's* tile is above its goal location, all variable tiles are no longer assignable. Thus, we can only give an assignment to n variable tiles. The issue remains that we can assign both a positive and negated variable tile, i.e., t_i and $\neg t_i$, and skip the assignment of some other variable tile t_j. The *south forcer gadget* ensures this can not happen by requiring south steps to be made at certain times in order to satisfy the reconfiguration requirements. This restriction ensures that each variable must receive a true or false assignment.

Thus, the generated instance of the reconfiguration problem is solvable if and only if the given 3SAT formula is solvable. Along with Thm. 4.3, it follows that the problem is NP-Complete when limited to two directions. □

7. Conclusion and Future Work

In this paper we investigated the Occupancy, Relocation, and Reconfiguration problems in an extremely limited variation of the tilt model. We discovered that even with these limitations, the problems of Relocation and Reconfiguration were still NP-Complete under the step transformation. Future work may focus on pushing the limitations even further; possibly considering relocation and/or reconfiguration of a rectangular board.

References

[1] Balanza-Martinez, J., Caballero, D., Cantu, A., Garcia, L., Luchsinger, A., Reyes, R., Schweller, R. and Wylie, T.: Full Tilt: Universal Constructors for General Shapes with Uniform External Forces, *Proc. 30th Annual ACM-SIAM Symposium on Discrete Algorithms*, *SODA'19* (2019).

[2] Balanza-Martinez, J., Caballero, D., Cantu, A.A., Flores, M., Gomez, T., Luchsinger, A., Reyes, R., Schweller, R. and Wylie, T.: Hierarchical Shape Construction and Complexity for Slidable Polyominoes under Uniform External Forces, *Proc. 31st Annual ACM-SIAM Symposium on Discrete Algorithms*, *SODA'20* (2020).

[3] Balanza-Martinez, J., Caballero, D., Cantu, A.A., Gomez, T., Luchsinger, A., Schweller, R., and Wylie, T.: Relocation with Uniform External Control in Limited Directions, *The 22nd Japan Conference on Discrete and Computational Geometry, Graphs, and Games*, *JCDCGGG*, pp.39–40 (2019).

[4] Becker, A., Habibi, G., Werfel, J., Rubenstein, M. and McLurkin, J.: Massive uniform manipulation: Controlling large populations of simple robots with a common input signal, *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.520–527 (online), DOI: 10.1109/IROS.2013.6696401 (2013).

[5] Becker, A., Demaine, E.D., Fekete, S.P., Habibi, G. and McLurkin, J.: Reconfiguring Massive Particle Swarms with Limited, Global Control, *Algorithms for Sensor Systems*, pp.51–66, Springer Berlin Heidelberg (2014).

[6] Becker, A.T., Demaine, E.D., Fekete, S.P., Lonsford, J. and Morris-Wright, R.: Particle computation: Complexity, algorithms, and logic, *Natural Computing*, Vol.18, pp.181–201 (2019).

[7] Becker, A.T., Demaine, E.D., Fekete, S.P. and McLurkin, J.: Particle computation: Designing worlds to control robot swarms with only global signals, *Proc. IEEE International Conference on Robotics and Automation*, *ICRA'14*, pp.6751–6756 (online), DOI: 10.1109/ICRA.2014.6907856 (2014).

[8] Becker, A.T., Fekete, S.P., Keldenich, P., Krupke, D., Rieck, C.,

Scheffer, C. and Schmidt, A.: Tilt Assembly: Algorithms for Micro-factories That Build Objects with Uniform External Forces, *Algorithmica* (online), DOI: 10.1007/s00453-018-0483-9 (2018).

[9]   Keldenich, P., Manzoor, S., Huang, L., Krupke, D., Schmidt, A., Fekete, S.P. and Becker, A.T.: On Designing 2D Discrete Workspaces to Sort or Classify Polynminoes, *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp.1–9 (online), DOI: 10.1109/IROS.2018.8594150 (2018).

[10]   Manzoor, S., Sheckman, S., Lonsford, J., Kim, H., Kim, M.J. and Becker, A.T.: Parallel Self-Assembly of Polyominoes Under Uniform Control Inputs, *IEEE Robotics and Automation Letters*, Vol.2, No.4, pp.2040–2047 (online), DOI: 10.1109/LRA.2017.2715402 (2017).

[11]   Schmidt, A., Manzoor, S., Huang, L., Becker, A.T. and Fekete, S.P.: Efficient Parallel Self-Assembly Under Uniform Control Inputs, *IEEE Robotics and Automation Letters*, Vol.3, pp.3521–3528 (2018).

David Caballero is a graduate student pursuing a Master's in Computer Science from University of Texas Rio Grande Valley. Research areas include Molecular Computation, Computational Complexity and Robot Motion Planning.

Angel A. Cantu is a Computer Scientist at the Southwest Research Institute for the Defense and Intelligence Division. His work involves research in geolocation, signals intelligence, and radar systems.

Timothy Gomez is a graduate student pursuing a Master's in Computer Science from University of Texas Rio Grande Valley. Research areas include Molecular Computation, Computational Complexity and Robot Motion Planning.

Austin Luchsinger is working as a graduate research assistant while pursuing a PhD in Electrical and Computer Engineering at the University of Texas at Austin. His research is primarily in the areas of molecular computation, robot motion planning, and self-assembly.

Robert Schweller is a Professor of Computer Science at the University of Texas Rio Grande Valley. Research interests include Algorithms, Molecular Computation, Computational Geometry, and Robot Motion Planning.

Tim Wylie is an Associate Professor of Computer Science at the University of Texas Rio Grande Valley. His research interests include Algorithms, Game Complexity, Molecular Computation, and Computational Geometry.