RELOCATION WITH UNIFORM EXTERNAL CONTROL IN LIMITED DIRECTIONS

Jose Balanza-Martinez, David Caballero, Angel A. Cantu, Timothy Gomez, Austin Luchsinger, Robert Schweller, Tim Wylie
Department of Computer Science, University of Texas - Rio Grande Valley, Edinburg, TX 78539, USA

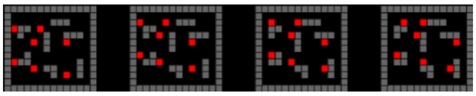
Abstract

We study a model where particles exist within a board and move single units based on uniform external forces. We investigate the complexity of deciding whether a single particle can be relocated to another position in the board, and whether a board configuration can be transformed into another configuration. We prove that the problems are NP-complete with 1 × 1 particles even when only allowed to move in 2 or 3 directions.

1 Introduction

The tilt model, proposed by Becker et al. [3], has foundations in classical motion planning. A couple of natural problems that arise in these computational systems are those of relocation and reconfiguration. *Relocation* is the problem of whether a sequence of tilts exists to relocate a tile from location *a* to location *b*. *Reconfiguration* asks if a sequence of tilts exists to transform board *A* to board *B* (specifying the location of all tiles). These were shown to be PSPACE-complete in 4-directions if a single polyomino larger than a 1 × 1 is in the system [1]. Here, we discuss a variant of this model (introduced in [2]) where particles exist within a board and move, in uniform, single unit distances (rather than maximally) in any of the four cardinal directions via an external force. These particles move in said direction unless the path is blocked by some "concrete" space. Figure 1 shows a simple example. We study these questions with only 1 × 1 tiles with limited usable directions (e.g. only tilting down and right). We further look at the complexity of the board geometry. A specific class of boards used for our relocation problem is "*x/y*-monotone", which can also be called vertically/horizontally monotone.

Definitions. A *board B* is a rectangular region in $\mathbb{Z}^2$ where positions are either *open* or *blocked*, meaning no tile can be on this location. A *tile/robot/particle* is a 1 × 1 polyomino with a label and a position at some location in *B*. A *configuration C* = (*B,P*) is the board and the set of tiles with their locations. A *step* is a cardinal direction command *d* = {*N,S,E,W*} that transforms one configuration into another by translating all tiles a unit distance in that direction unless the adjacent location is blocked or occupied by another tile. A *step sequence* is a series of steps which can be inferred from a series of directions D = hd_1,...,d_ki where each $d_i \in D$ implies a step in that direction.



(a) Init **(b)** h*N*i **(c)** h*E*i **(d)** h*E*i

Figure 1: An example step sequence h*N,E,E*i. (a) The initial board configuration. (b) The resultant configuration after an *N* step. (c) The resultant configuration after an *E* step. (d) The final configuration after one more *E* step.

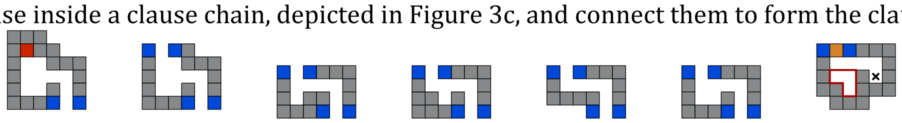
We also note that both problems are in NP, since with limited directions, there is a limited number of possible steps before the configuration cannot change, or can only move between a small number of configurations.

2 Relocation

Here we show the relocation problem is hard with limited directions (two or three) even in a monotone board via a reduction from 3-SAT.

Theorem 1. *Given a monotone board, the relocation problem in the single step model is NP-complete when limited to two or three directions.*

The 3-SAT reduction utilizes gadgets that have *binding locations* on their top and bottom locations which is how the gadgets connect to form *chains*. There are 3 chain gadgets: the *literal chain*, *clause chain*, and *validation chain*. They connect at their binding locations to form one chain representing the 3-SAT formula. Every literal is represented as one tile, and trapping it in a certain location on the board sets that literal to 'true'. We place 3 literal tiles for every clause inside a clause chain, depicted in Figure 3c, and connect them to form the clause chain.

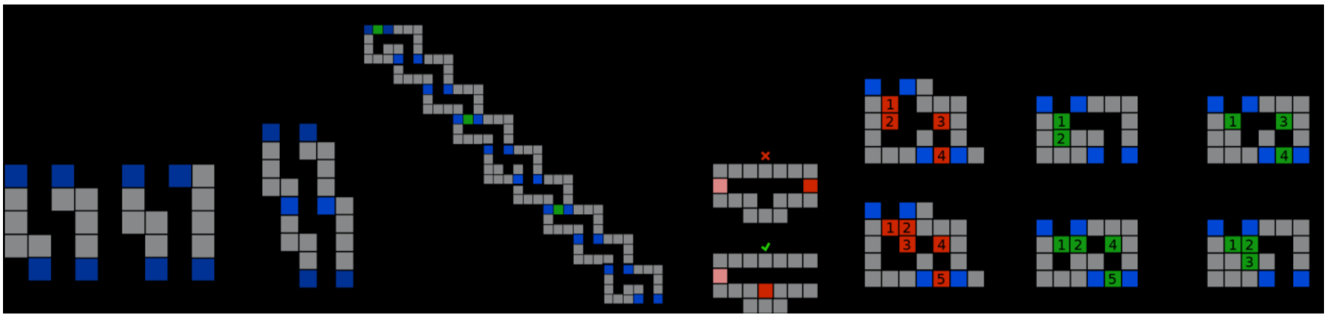


(a) **(b)** **(c)** **(d)** **(e)** **(f)** **(g)**

Figure 2: Different individual gadgets.

The tile in Figure 2a will be relocated to the gadget in Figure 2g, which is the last gadget in the validation chain. The validation chain contains tiles that force 'appropriate' assignments of the literal tiles– the validation tiles could block the relocation tile. Moreover, if any clause is not satisfied, the 'excess' tiles from the clause chain will enter the validation chain and block the relocation tile. The three chains force the proper assignment of literals and checks the satisfiability of all clauses, allowing the relocation tile to be placed only if all the clauses were satisfied.

1



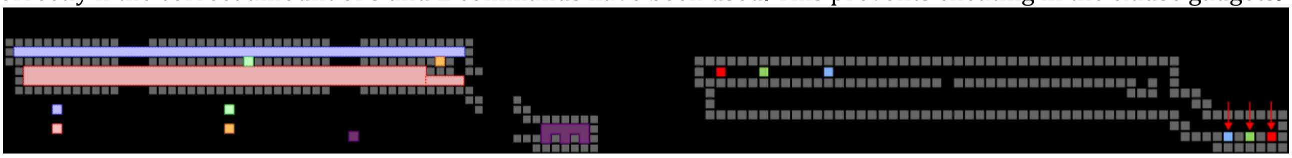
(a) **(b)** **(c)** **(d)** **(e)**

Figure 3: (a) Demonstration of two gadgets (b) binding at their bottom locations. (c) An example clause gadget. (d) Truth assignment of some literal tile. (e) A truth assignment for a variable x. The numbering on the tiles depict the step move order. Setting literal x_i to true is depicted in the upper three images, and setting it to false is depicted in the lower three.

3 Reconfiguration

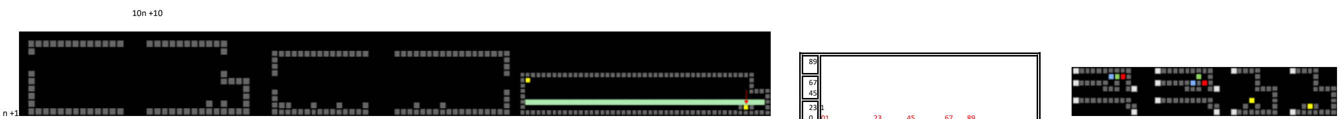
Reconfiguration is a variant of relocation that ensures we know where every tile on the board is located. The input to the problem is two configurations: an initial and final. We show that in two directions the problem is NP-complete via a 3-SAT reduction. We will use South and East as the directions. Given a 3-SAT formula consisting of variables $\{x_1,x_2,x_3,...,x_n\}$, separated into m clauses of the form $(A \vee B \vee C)$ where each A,B,C is of the form x_i or $\neg x_i$, we create an instance of the reconfiguration problem that is solvable if and only if the 3-SAT formula is satisfiable. **Theorem 2.** *The reconfiguration problem in the single step model is NP-complete when limited to two directions.*

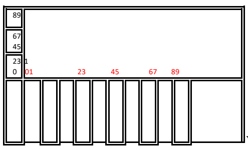
Figure 4a shows the basic structure of the clause gadgets and Figure 4b shows how variable tiles are placed. For the clause, the variables must be placed in reverse order in the three slots at the bottom right. This can only occur if one of the variables has a true value. Otherwise, one of the tiles will go into the wrong slot. In order to force the constraints, the other gadgets, shown in Figure 5, specify the location of other tiles which can only be placed correctly if the correct amount of S and E commands have been used. This prevents cheating in the clause gadgets.



(a) **(b)**

Figure 4: (a) 3-SAT clause gadget, where n is the total number of unique variables, and m indicates that this clause is the m^{th} clause in the formula. (b) Example of variable placement for clause $(x_1 \vee \neg x_2 \vee x_3)$ in a 5-variable formula. Goal locations indicated by red arrows





(a) **(b)** **(c)** **(d)** **(e)**

Figure 5: (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. (c) Example of south limiter tile placement in a 5 variable formula. Goal location indicated by red arrow. Post assignment zone highlighted in green. (d) Example of south forcer tile placement in a 5 variable formula. Goal location indicated by red labels. (e) State of a clause gadget in which no variable tiles evaluated to a literal true before and after the forced south tilt, and the respective states of the *south limiter gadget*.

Acknowledgments. This research was supported in part by National Science Foundation Grant CCF-1817602.

References

[1] Jose Balanza-Martinez, David Caballero, Angel Cantu, Luis Garcia, Austin Luchsinger, Rene Reyes, Robert Schweller, and Tim Wylie. Full tilt: Universal constructors for general shapes with uniform external forces. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'19, 2019.

[2] Aaron Becker, Erik D. Demaine, Sa´ndor P. Fekete, Golnaz Habibi, and James McLurkin. Reconfiguring massive particle swarms with limited, global control. In *Algorithms for Sensor Systems*, pages 51–66, 2014.

[3] Aaron T. Becker, Erik D. Demaine, Sa´ndor P. Fekete, and James McLurkin. Particle computation: Designing worlds to control robot swarms with only global signals. In *Proceedings of the IEEE International Conference on Robotics and Automation*, ICRA'14, pages 6751–6756, May 2014.