On Efficient Connectivity-Preserving Transformations in a Grid

Abdullah Almethen, Othon Michail and Igor Potapov

Department of Computer Science
University Of Liverpool

16th International Symposium on Algorithms for Sensor Systems, ALGOSENSORS 2020

Outline

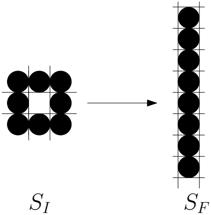
1. Introduction
2. Our Contribution
3. Definitions
4. Transformations
5. Conclusions

Introduction

- Recent developments for collective robotic systems.
- From the scale of milli or micro down to nano size of individual devices.
- The research area of *programmable matter* - materials that can algorithmically change its physical properties:
  - such as their shape, colour, conductivity and density.
- The implementation indicates whether the system is:
  - centralised.
  - decentralised.
- The need for the development of an algorithmic theory of such systems.

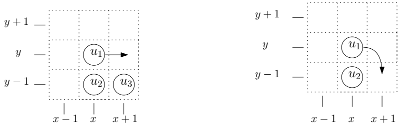
Settings

- 2D square grid.
- Each cell is occupied by a distinct device (node) on the grid.
- A number of nodes connected to each other, forming a shape S_I.
- Given a desired target shape S_F of the same order.
- **Goal:** transform S_I into S_F via a finite number of valid moves.



Models of individual moves

- Only one node moves in a single time-step.
- Such as, Dumitrescu *et al.* IJRR'04 and Michail *et al.* JCSS'19:
  - An individual device can move over and turn around its neighbours through empty space.



- Akitaya *et al.* ESA'19 consider transformations based on similar moves.
- To transform some pairs of connected shapes, $\Omega(n^2)$ moves are required for all models of constant-distance individual moves.

Cont.

-**This motivates the study of alternative types of moves that are reasonable with respect to practical implementations and allow for sub-quadratic reconfiguration time in the worst case.**

-**There are attempts to provide alternatives for more efficient reconfiguration.**

Parallel transformations

- Multiple nodes move together in a single time-step.
- Especially in distributed systems.
  - nodes can make independent decisions and move locally in parallel to other nodes.
- Theoretical studies, such as Daymude *et al.*, Natural Computing'18.
- Practical implementations, such as Rubesntein *et al.*, Science'14.
- It can be shown that a connected shape can transform into any other connected shape, by performing in the worst case $O(n)$ parallel moves around the perimeter of the shape.

Models of more powerful mechanism

- Equip nodes with strong actuation mechanisms.
- Reduce the inherent distance by a factor greater than a constant in a single time-step.
- Linear-strength mechanisms, such as:
  - Aloupis *et al.*, (Computational Geometry'13) provide a node with arms that are capable to extend and contract a neighbour.
  - Woods *et al.*, (ITCS'13) proposed an alternative linear-strength mechanism, where a node has the ability to rotate a whole line of consecutive nodes.
  - Czyzowicz *et al.*, (ESA'19) consider a single moving robot that transforms a static shape by carrying its tiles one at a time.
  - Recently, Almethen *et al.*, (TCS'20) introduce the line-pushing model.

The line-pushing model

[Almethen et al., TCS'20]:

- A new linear-strength mechanism, where a whole line of consecutive devices can, in a single time-step, move by one position in a given direction.
- Generalisation of other existing models of reconfiguration:
  - Focus on exploiting the power of parallelism.
- Pure theoretical interest.
- A practical framework.

The line-pushing model

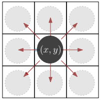
- Simulate the rotation and sliding models.
- All transformations of individual nodes, their universality and reversibility properties still hold true in the line-pushing model.
- Achieved sub-quadratic time transformations,
  - An $O(n \log n)$-time universal transformation which does not preserve connectivity.
  - A connectivity-preserving $O(n\sqrt{n})$-time transformation for the special case of transforming a diagonal into a straight line.

Our Contribution

- We build upon the findings of Almethen *et al.*, TCS'20.
- All transformations in the present study preserve connectivity of the shape during the transformations.
- An $O(n \log n)$-time connectivity-preserving transformations in which the *associated graphs* of (S_I, S_F) are isomorphic to a Hamiltonian line (defined later).
  - Asymptotically equal to the best known running time of connectivity-breaking transformation.
- An $O(n\sqrt{n})$-time connectivity-preserving universal transformation.

Preliminaries

- Each node is connected to a neighbour vertically, horizontally or diagonally.



A node is *connected* to a neighbour at any directions.

- A line L is a sequence of nodes occupying consecutive cells in one direction of the grid, that is, either vertically or horizontally but not diagonally.

- Each node is equipped with a linear-strength mechanism which enables it to move a whole line in a single time-step.

Definitions

- A **line move** is an operation of moving all nodes of L together in a single time-step towards a position adjacent to one of L's endpoints, in a given direction d of the grid, $d \in \{up, down, right, left\}$.

Definition (A permissible line move)

A line $L = (x, y), (x+1, y), \dots, (x+k-1, y)$ of length k, where $1 \leq k \leq n$, can push all its k nodes rightwards in a single move to positions $(x+1, y), (x+2, y), \dots, (x+k, y)$ iff there exists an empty cell at $(x+k, y)$. The "down", "left", and "up" moves are defined symmetrically, by rotating the whole shape 90°, 180° and 270° clockwise, respectively.

- **The problem**:
  → Transform S_I into S_F via a finite number of valid line moves.

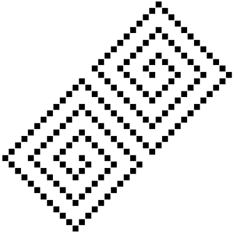
Definitions

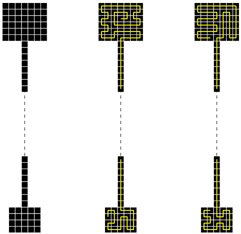
Definition

A graph $G(S) = (V, E)$ is *associated* with a shape S, where $u \in V$ iff u is a node of S and $(u, v) \in E$ iff u and v are neighbours in S.

Definition (Hamiltonian Shapes)

A shape S is called Hamiltonian iff $G(S) = (V, E)$ is isomorphic to a Hamiltonian path, i.e., a path starting from a node $u \in V$, visiting every node in V exactly once and ending at a node $v \in V$, where $v \neq u$. $\mathcal{H}$ denotes the family of all Hamiltonian shapes.





Definitions

- A configuration of the system is defined as a mapping $C\colon \mathbb{Z} \times \mathbb{Z} \to \{0, 1\}$, where $C(x, y) = 0$ if cell(x, y) is empty or $C(x, y) = 1$ if cell(x, y) is occupied by a node.

- A *rectangular path* P over the set of cells is defined as $P = [c_1, c_2, c_3, \ldots, c_k]$, where $c_i, c_{i+1} \in \mathbb{Z} \times \mathbb{Z}$ are two cells adjacent to each other either vertically or horizontally, for all $i \in \{1, 2, \ldots, k-1\}$.

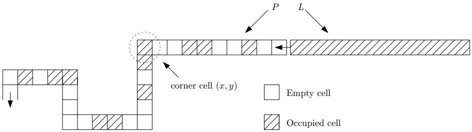
- Given any P, let C_P be the configuration of P defined as the subset of C (configuration of the system) restricted to the cells of P.

Transparency of Line Moves

Proposition

Let S be any shape, $L \subseteq S$ any line and P a rectangular path starting from a position adjacent to one of L's endpoints. There is a way to move L along P, while satisfying all the following properties:

1. **No delay:** *The number of steps is asymptotically equal to that of an optimum move of L along P in the case of C_P being empty (i.e., if no cells were occupied). That is, L is not delayed, independently of what C_P is.*
2. **No effect:** *After L's move along P, $C'_P = C_P$, i.e., the cell configuration has remained unchanged. Moreover, no occupied cell in C_P is ever emptied during L's move (but unoccupied cells may be temporarily occupied).*
3. **No break:** *S remains connected throughout L's move.*



Problem Definitions

HAMILTONIANCONNECTED. Given a pair of connected Hamiltonian shapes (S_I, S_F) of the same order, where S_I is the initial shape and S_F the target shape, transform S_I into S_F while preserving connectivity throughout the transformation.

DIAGONALTOLINECONNECTED. A special case of HAMILTONIANCONNECTED in which S_I is a diagonal line and S_F is a straight line.

UNIVERSALCONNECTED. Given *any* pair of connected shapes (S_I, S_F) of the same order, where S_I is the initial shape and S_F the target shape, transform S_I into S_F while preserving connectivity throughout the transformation.

An $O(n \log n)$-time Transformation for Hamiltonian Shapes

- Called *Walk-Through-Path* and solves HamiltonianConnected.

- Starts from one endpoint of the Hamiltonian path of S_I and applies a recursive successive doubling technique to transform S_I into a straight line S_L.

- Replace S_I with S_F reversely in *Walk-Through-Path* to go from S_I to S_F in the same asymptotic time.

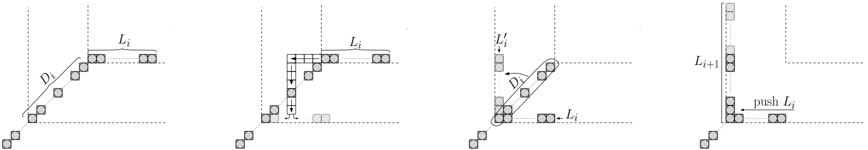


Figure: A snapshot of phase i of *Walk-Through-Path* applied on a diagonal. Light grey cells represent the ending positions of the corresponding moves depicted in each sub-figure.

An $O(n \log n)$-time Transformation for Hamiltonian Shapes

Algorithm 1: HamiltonianToLine(S)

$S = (u_0, u_1, \ldots, u_{|S|-1})$ is a Hamiltonian shape
Initial conditions: $S \leftarrow S_I$ and $L_0 \leftarrow \{u_0\}$

```
for i = 0, ..., log |S| do
    LineWalk(L_i)
    S_i ← select(2^i) // select the next terminal subset of 2^i consecutive nodes of S
    L'_i ← HamiltonianToLine(S_i) // recursive call on S_i
    L_{i+1} ← combine(L_i, L'_i) // combines L_i and L'_i into a new straight line L_{i+1}
end
```

Output: a straight line S_L

- Main challenge: make the above transformation work in the general case.
- Hamiltonian shapes do not necessarily provide free space.
- Moving a line through the remaining configuration of nodes.
- Does not break their and its own connectivity by *LineWalk* operation.

An $O(n \log n)$-time Transformation for Hamiltonian Shapes

Theorem

For any pair of Hamiltonian shapes $S_I, S_F \in \mathcal{H}$ of the same order n, Walk-Through-Path transforms S_I into S_F (and S_F into S_I) in $O(n \log n)$ moves, while preserving connectivity of the shape during its course.

$$T = \sum_{i=1}^{\log n} T(i) = \sum_{i=1}^{\log n} 2^{i-1}(i-1) - 2^i = \sum_{i=1}^{\log n - 1} (i-2)2^i - 2^{\log n} \leq \sum_{i=1}^{\log n - 1} i \cdot 2^i - n$$

$$\leq \sum_{j=1}^{\log n} \sum_{i=j}^{\log n} 2^i - n \ le \sum_{j=1}^{\log n} n - n \leq n \log n - n \leq O(n \log n).$$

An $O(n\sqrt{n})$-time Universal Transformation

- **Solves the** UniversalConnected **problem and is called UC-Box.**

- First, Compute a spanning tree T of the associated graph $G(S_I)$ of S_I.

- Enclose S_I into an $n \times n$ square box and divide it into $\sqrt{n} \times \sqrt{n}$ square sub-boxes.

- Each occupied sub-box contains one or more maximal sub-trees of T.

- Each such sub-tree corresponds to a sub-shape of S_I, called a *component*.

- Pick a leaf sub-tree T_l which is associated with component C_l occuping sub-box B_l.

- B_p is the sub-box adjacent to B_l containing the unique parent sub-tree T_p of T_l.

- Compress all nodes of C_l into B_p while keeping the nodes of C_p (the component of T_p) within B_p.

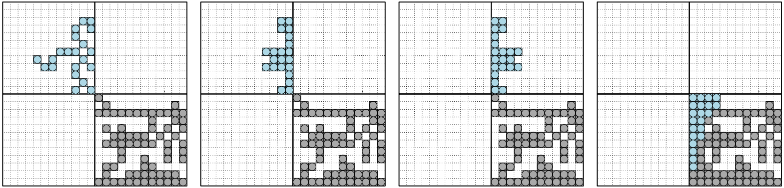
Algorithm 2: COMPRESS(S)

$S = (u_1, u_2, \ldots, u_{|S|})$ is a connected shape, T is a spanning tree of $G(S)$ **repeat**

```
    C_l ← pick(T_l) // select a leaf component associated with a leaf sub-tree
    Compress(C_l) // start compressing the leaf component
    if C_l collides then
        C'_r ← combine(C_r, C_l) or C'_m ← combine(C_m, C_l) // as described in text
    else
        C'_p ← combine(C_p, C_l) // combine C_l with a parent component
    end
    update(T) // update sub-trees and remove cycles after compression
```

until *the whole shape is compressed into a* $\sqrt{n} \times \sqrt{n}$ *square*

Output: a square shape S_C



An $O(n\sqrt{n})$-time Universal Transformation

- Main technical challenge: make this strategy work universally.
- S_I might occupy several sub-boxes of different configurations.
- Preserving connectivity during the transformation.
- We manage to upper bound the cost of each charging phase independently of the order of compressions.

Theorem

For any pair of connected shapes (S_I, S_F) of the same order n, UC-Box transforms S_I into S_F (and S_F into S_I) in $O(n\sqrt{n})$ steps, while preserving connectivity during its course.

Open Problems

- An $O(n \log n)$-time universal connectivity-preserving transformation.
- A general $\Omega(n \log n)$-time matching lower bound.
- A centralised parallel version in which more than one line can be moved concurrently in a single time-step.
- A distributed version of the parallel model.
  - The nodes operate autonomously through local control and under limited information.