On Geometric Shape Construction via Growth Operations

ALGOSENSORS 2022, 8th Sep 2022, Potsdam, Germany

Nada Almalki and Othon Michail

Department of Computer Science,
University of Liverpool, UK

UNIVERSITY OF LIVERPOOL

Introduction and Motivation

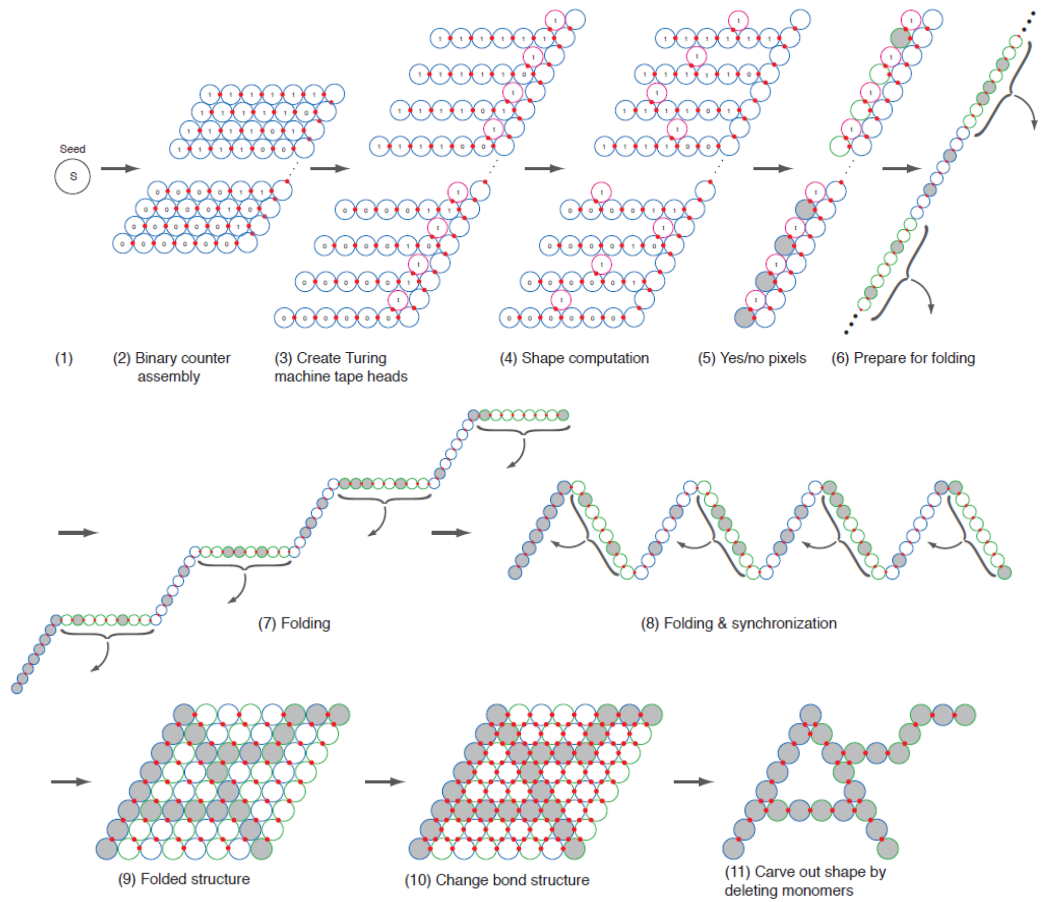
1. Woods *et al.* model

- How to generate connected 2D geometric shapes in polylogarithmic time
- Develop lines, squares and some arbitrary shapes

2. Mertzios *et al.* model

- Network-level abstraction of programmable matter systems
- Self-replication and local reconfiguration

[Mertzios *et al.*, ALGOSENSORS'22]



Source: [Woods *et al.*, ITCS'13]

UNIVERSITY OF LIVERPOOL

Contribution

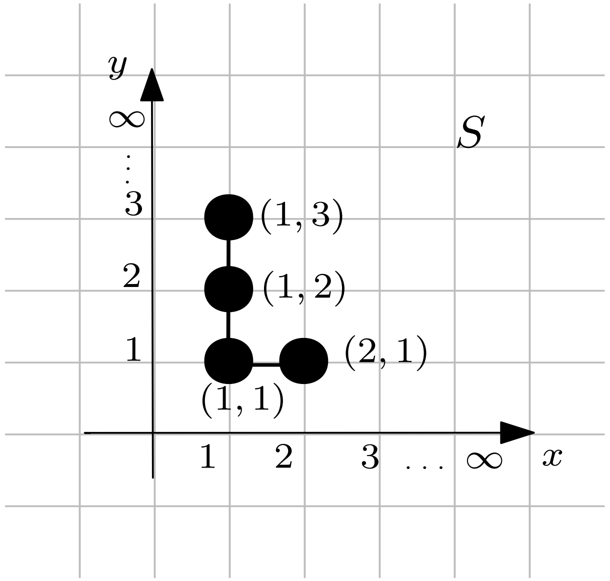
We study three growth operations in a **centralized** **geometric** setting:

1. ***Full doubling:***
   - Characterize the structure of the class of shapes that are reachable from any S_I
2. ***RC doubling:***
   - There is a **linear-time** centralized algorithm that for any pair of shapes (S_I, S_F) decides if S_F can be constructed from S_I
   - If yes, returns an **$O(\log n)$**-time-step constructor
3. ***Doubling:***
   - Some shapes cannot be constructed in **sub-linear** time-steps
   - There are **two universal constructors** of any S_F from a singleton S_I
   - Both constructors can be computed by polynomial-time centralized algorithms for any shape S_F

UNIVERSITY OF LIVERPOOL

Our Model

- **Geometric** version of the abstract network-growth model of Mertzios *et al.*
- *n nodes* that form a connected shape *S*
- The considered model operates on a **2D square** grid
- Time consists of **discrete** time-steps
- Additional **properties** of growth operations:
  - Parallel operations
  - Single-direction growth operations



Note (two distinct notions):

- Time-steps of a growth operation **(*time-steps*)**
- The running time of a centralized algorithm **(*time*)**

Problems

1. Class characterization

What is the **class of shapes** that can be constructed efficiently from a given initial shape via a sequence of growth operations?

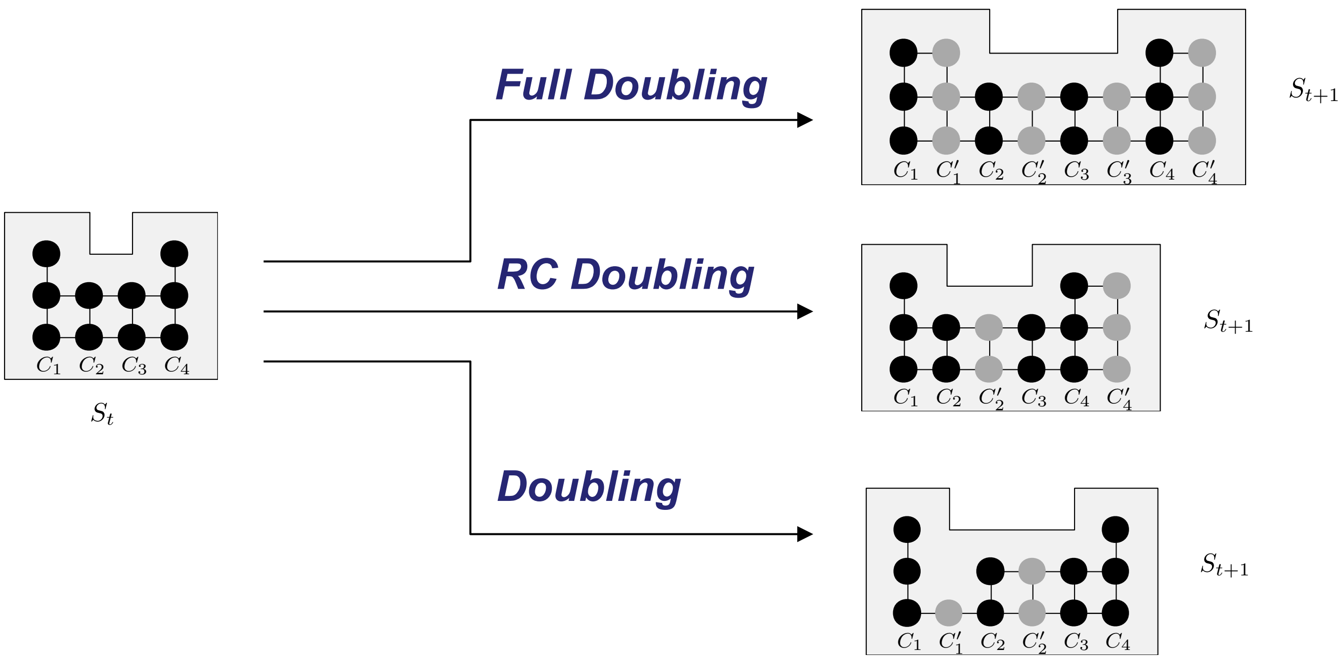
2. SHAPECONSTRUCTION

Is there a **polynomial-time** centralized algorithm that can decide if a given target shape S_F can be constructed from a given initial shape S_I and, whenever the answer is positive, return an **efficient constructor** of S_F from S_I?

UNIVERSITY OF LIVERPOOL

Growth Operations

A **growth operation *o*** is an operation that when applied on a shape instance S_t, for all time-steps $t \geq 0$, yields a new shape instance $\boldsymbol{S_{t+1}} = \boldsymbol{o(S_t)}$, such that $|\boldsymbol{S_{t+1}}| > |\boldsymbol{S_t}|$.



Full Doubling Operation

Reconfiguration Function

A reconfiguration function $F_{l,k}$ maps a shape to another shape in **two Phases:**

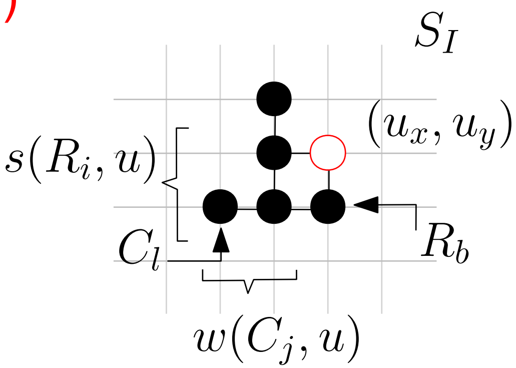
Phase 1

The coordinates of $|S|$ points of $F_{l,k}(S)$ are determined as a function of the coordinates of the points of S.

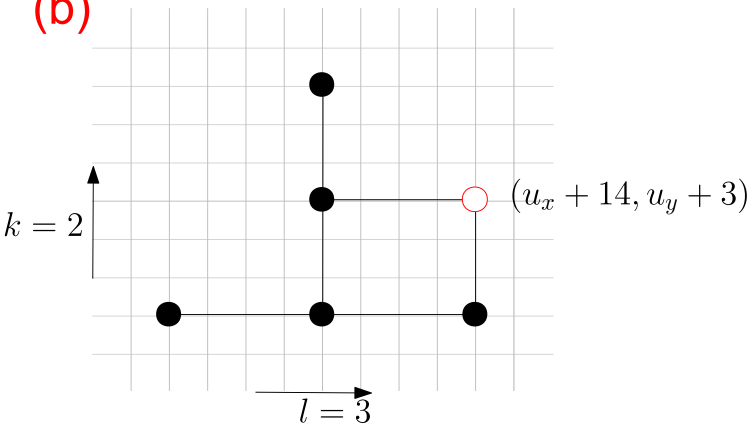
For each $u \in S$ the coordinates of $u' \in F_{l,k}(S)$ are given by

$(u_x + (2^l - 1)w(C_j, u), u_y + (2^k - 1)s(R_i, u))$.

(a)



(b)

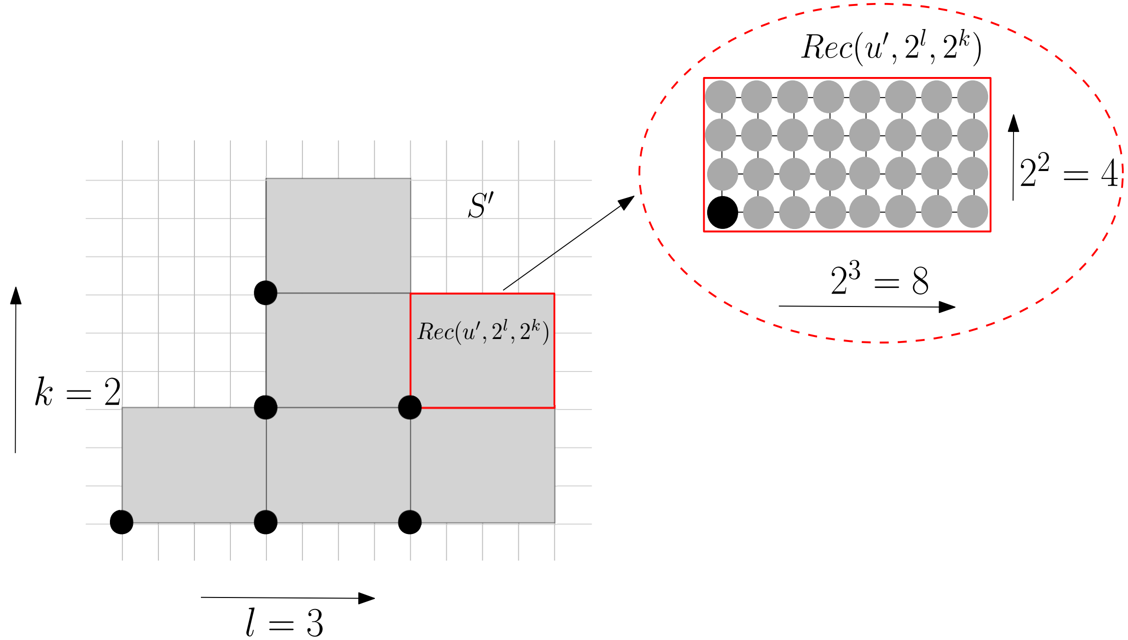


Full Doubling Operation

Reconfiguration Function

Phase 2

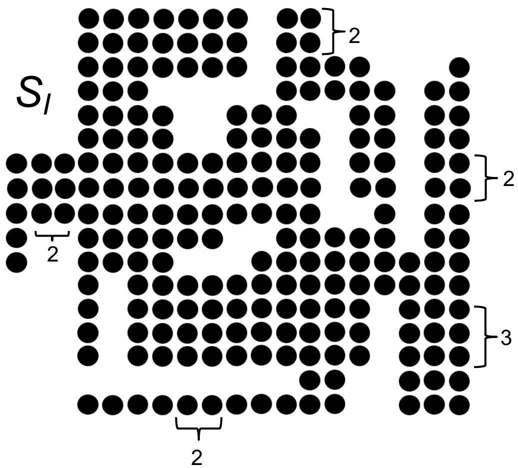
Generate the **Cartesian product** around u' such that, $Rec(u', 2^l, 2^k) = \{u'_x + 1, \ldots, u'_x + (2^l - 1)\} \times \{u'_y + 1, \ldots, u'_y + (2^k - 1)\}$ originating *at* u'.

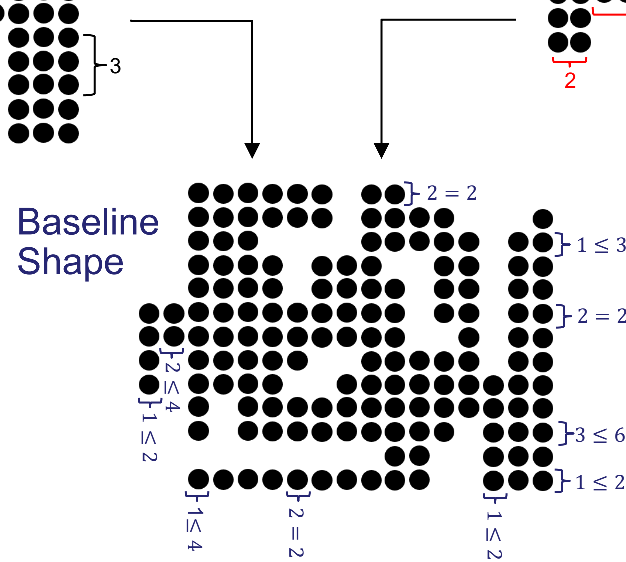


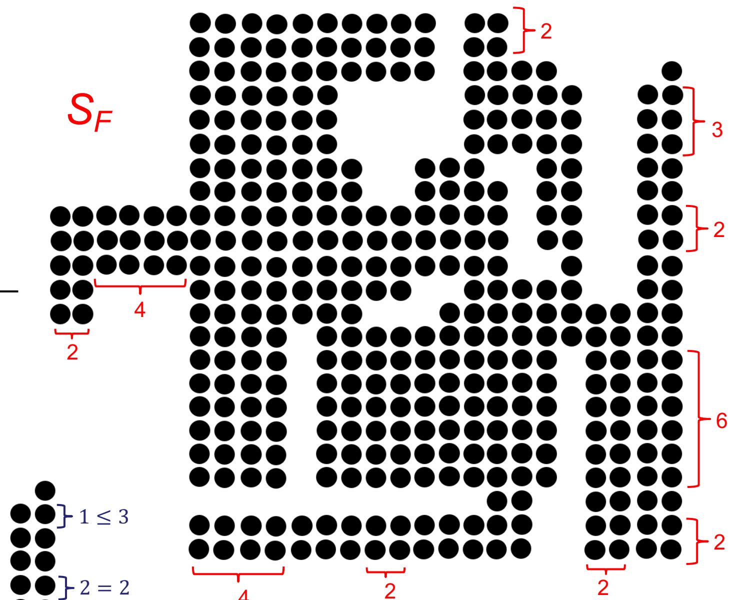
Theorem 1. Given any initial shape S_I and any sequence of l east and k north full doubling operations, the obtained shape is $S_F = F_{l,k}(S_I)$.

RC Doubling Operation

Theorem 2. A shape S_I can generate a shape S_F through a sequence of *RC* doubling operations iff $\boldsymbol{B(S_I) = B(S_F) = B}$ and for every column *C* and row *R* of *B* it holds that $\boldsymbol{M_{SF}(C) \geq M_{SI}(C)}$ and $\boldsymbol{M_{SF}(R) \geq M_{SI}(R)}$.







- $B(S_I) = B(S_F)$
- $M_{SF}(C) \geq M_{SI}(C)$
- $M_{SF}(R) \geq M_{SI}(R)$

RC Doubling Operation

Theorem 3. There is a **linear-time** algorithm for ***ShapeConstruction*** under *RC doubling* operations. In particular, given any pair of shapes (S_I , S_F), when ($S_I \rightsquigarrow S_F$) the algorithm returns a constructor σ of S_F from S_I of ***O*(log *n*)**-time-steps.

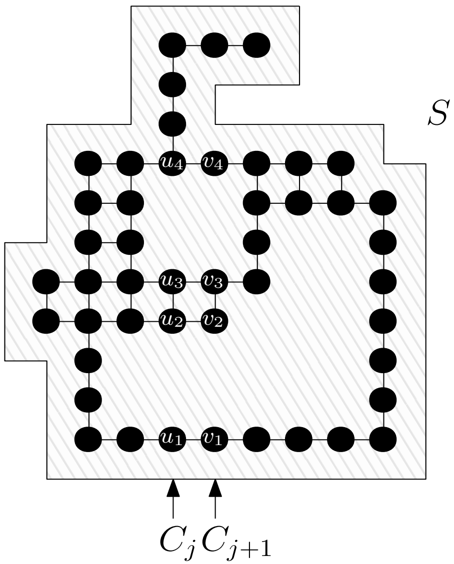
Algorithm 3: Constructor $S_I \overset{\sigma}{\rightsquigarrow} S_F$

Input : Decision, from Algorithm 2.
Output: Constructor σ.

while *Decision* **do**
    /* Find the constructor σ for columns. */
    for *every column* (C_j *of* S_F) **do**
        for *every column* (C_j *of* S_I) **do**
            if *the index of every column* C_j *of* S_F *and* S_I *are equal* **then**
                if *multiplicity of* $M_{S_F}(C_j)$ *are greater than* $M_{S_I}(C_j)$ **then**
                    compute the difference in $m(C_j)$ variable
                    append ($m(C_j)$ to σ)
compute the maximum of σ in max_{value} variable.
/* Count the steps k for doubling columns. */
if $k = \log max_{value}$ mod2 *does not equal* 0 **then**
    add one extra step to k variable to double the remaining columns that are not power of 2.
return $\sigma = \{m(C_1), m(C_2), \ldots, m(C_j)\}$.

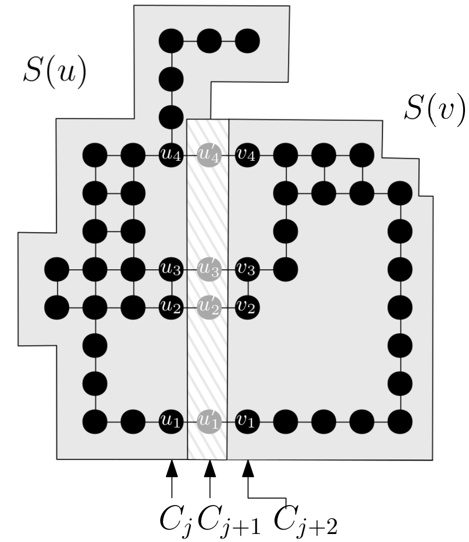
UNIVERSITY OF LIVERPOOL

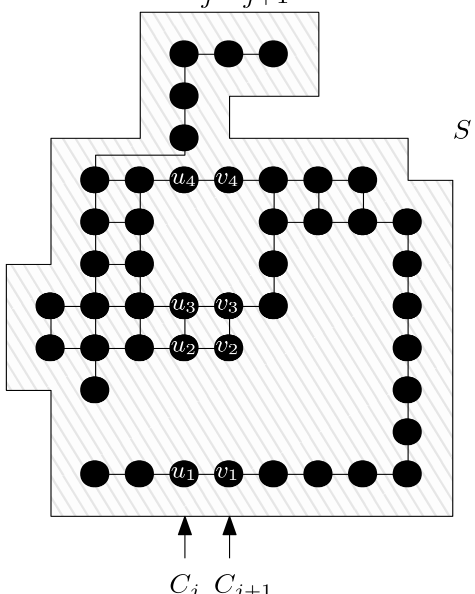
Doubling Operation



Rigidity-Preserving

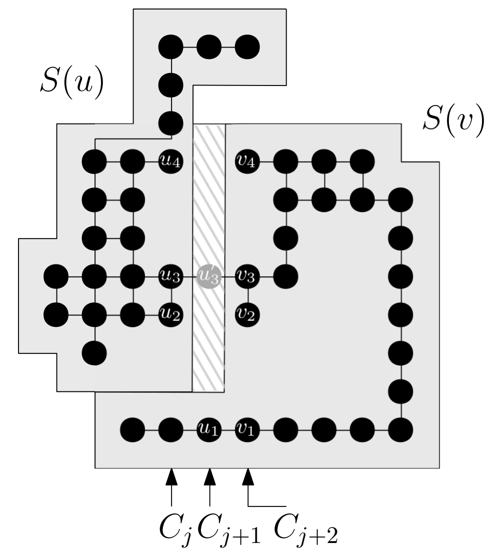
$I(C_j, u_1, \ldots, u_4)$





Rigidity-Breaking

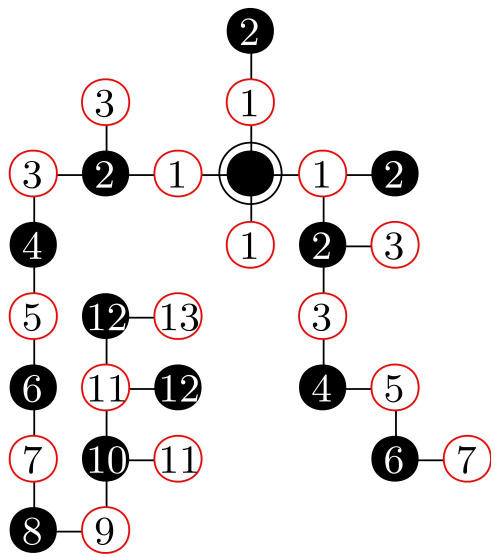
$I(C_j(u_3))$



UNIVERSITY OF LIVERPOOL

Doubling Operation

Proposition. For any shapes S_I and S_F, where $S_I \subseteq S_F$, there is a **linear** time-step constructor of S_F from S_I.



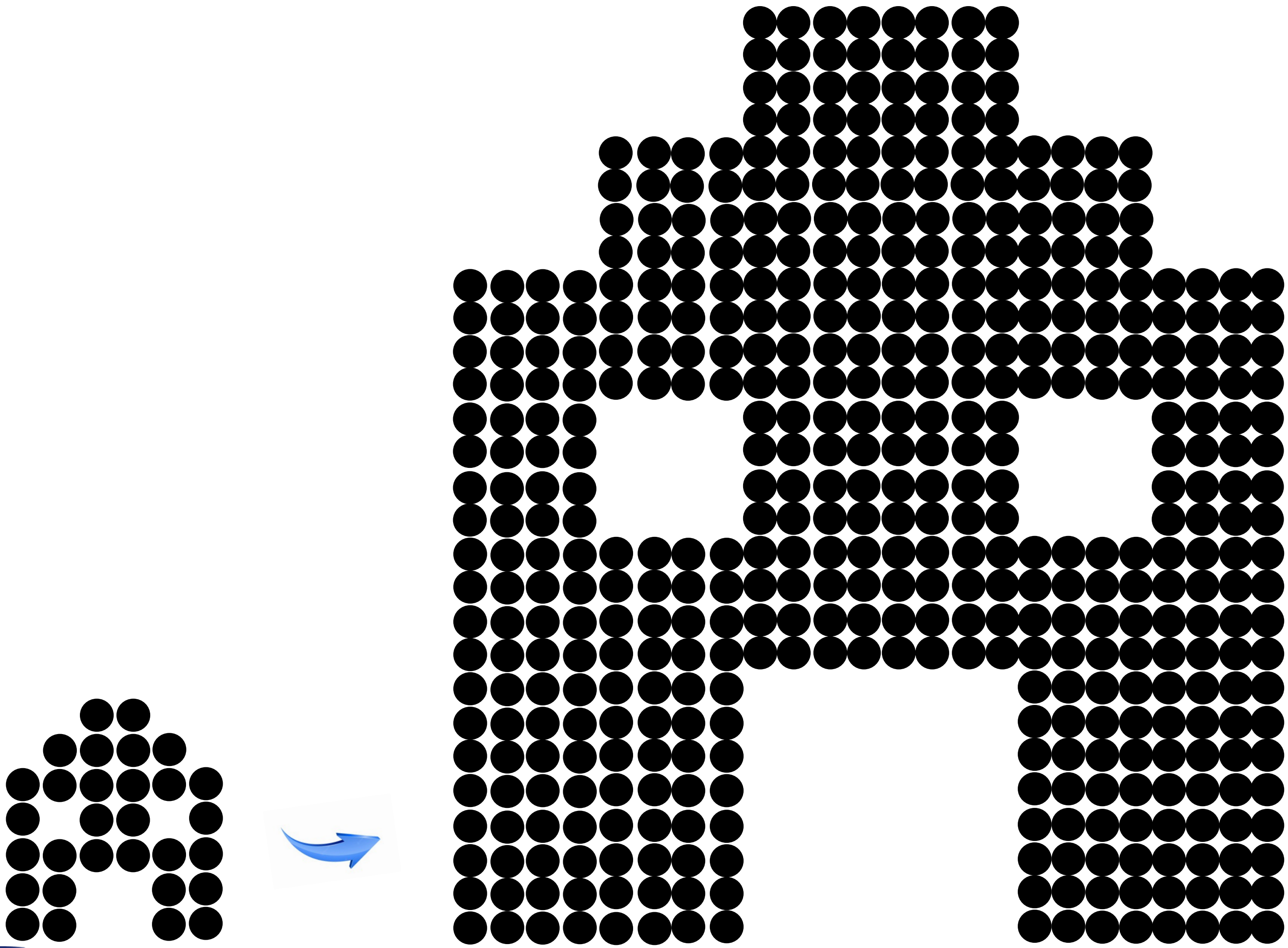
Proof

- Compute spanning tree ***T*** using **BFS**
- **Empty and adjacent** positions
- Fill all positions without **pushing existing nodes**

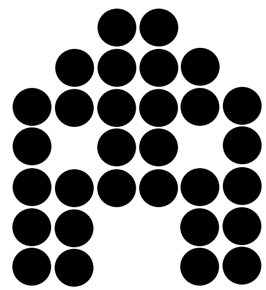
UNIVERSITY OF LIVERPOOL

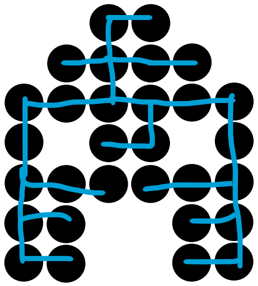
Doubling Operation

Theorem 4. Given any connected target shape S_F, there is an **[$O(|B(S_F)|) + O(\log |S_F|)$]**-time-step constructor of S_F from $S_I = \{u_0\}$ through doubling operations. Moreover, there is a polynomial-time algorithm computing such a constructor on every input S_F.



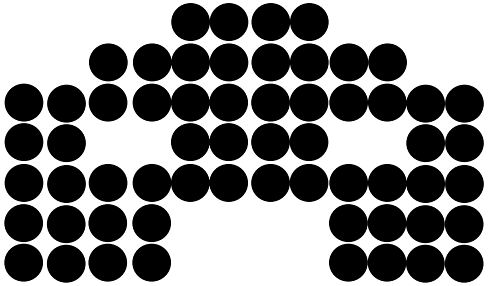
UNIVERSITY OF
LIVERPOOL

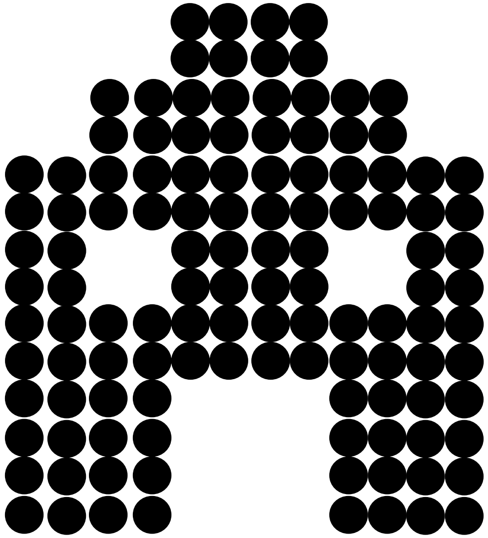


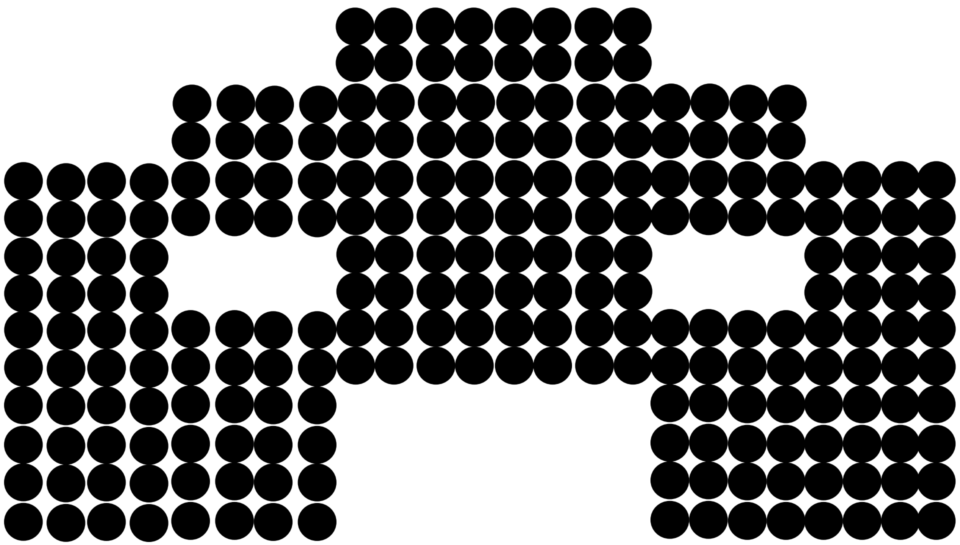


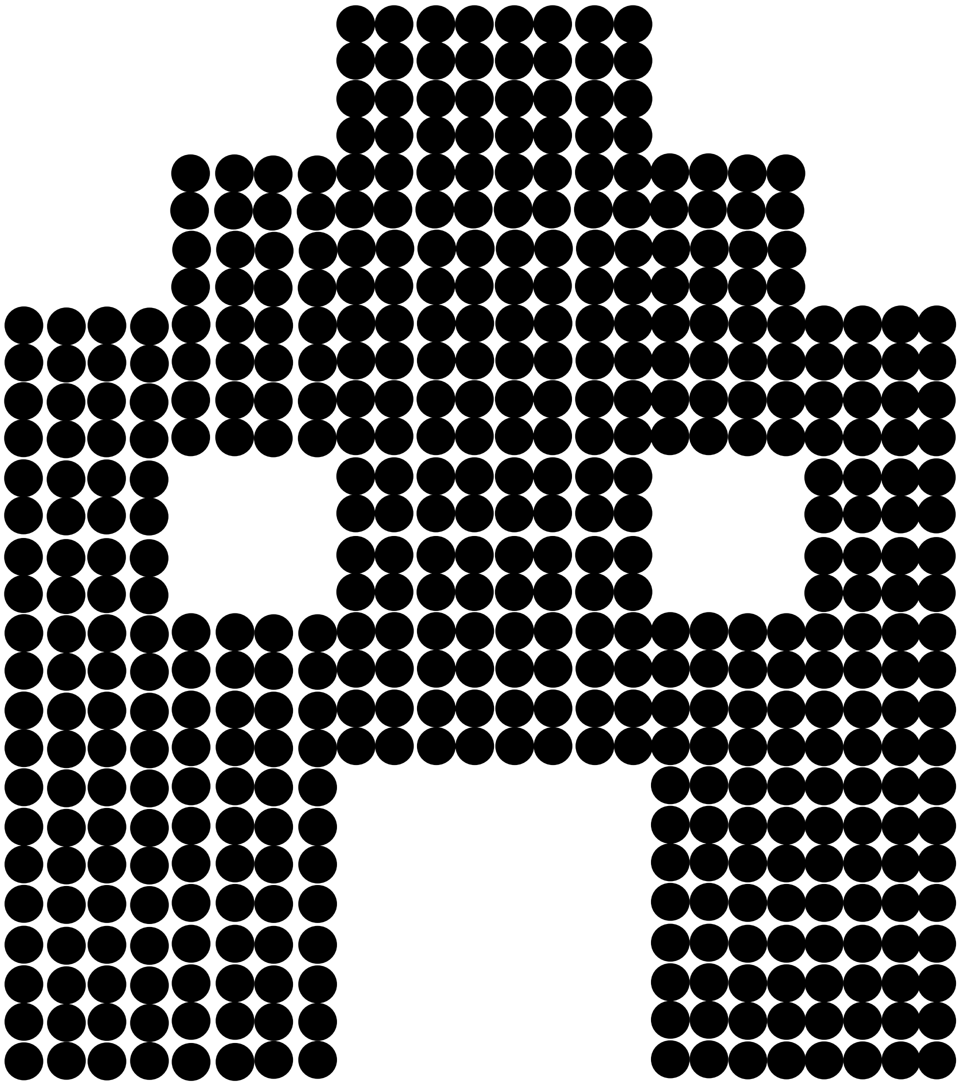
UNIVERSITY OF
LIVERPOOL

UNIVERSITY OF LIVERPOOL









UNIVERSITY OF
LIVERPOOL

Conclusion

- **Summary:**
  - Two problems were considered: **Class characterization** and **ShapeConstruction**
  - There is a **linear-time** centralized algorithm that for any pair of shapes (S_I, S_F) decides if S_F can be constructed from S_I, if yes, returns an ***O*(log *n*)-time-step constructor**
  - There are **two universal constructors** of any S_F from a singleton S_I

- **Future Research Directions:**
  - Obtain an **optimal** constructor
  - Explore our model in **3D** setting
  - Develop **distributed** version

UNIVERSITY OF LIVERPOOL

Any Questions?