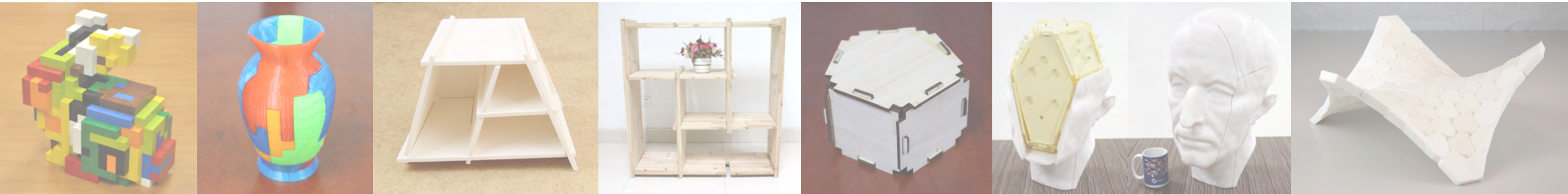
SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

CGL

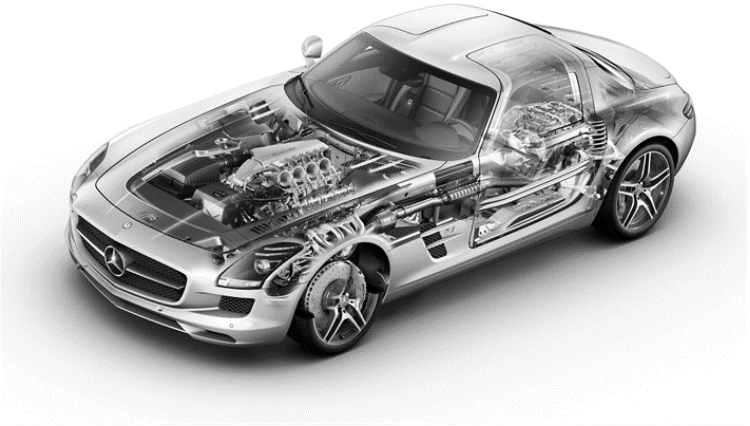
3D Interlocking Assemblies: Design and Applications

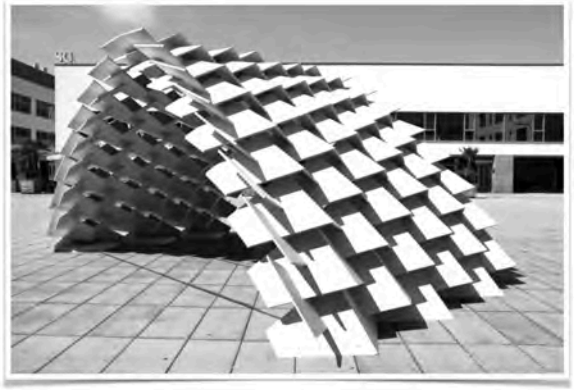
Peng SONG, SUTD



3D Assemblies

Composed of multiple component parts with a specific form and functionality





Steady Assembly

Need parts joining approach to restrict parts relative movements

Parts Joining

Unsteady

Irreversible

Break parts

Tedious

Interlocking Assembly

Can we make a steady assembly without relying on any fasteners?

Interlocking Assembly

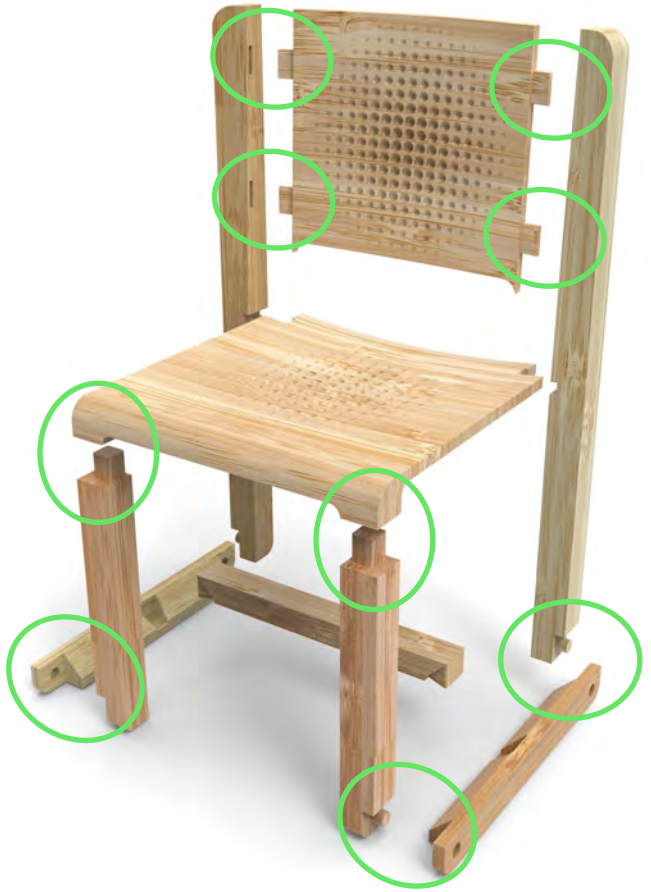
Can we make a steady assembly without relying on any fasteners?

Solution: parts connected based on their geometric arrangements.

Interlocking Assembly

Can we make a steady assembly without relying on any fasteners?

Solution: parts connected based on their geometric arrangements.



Intriguing Properties

Ease of assembly

Steady

Disassemblable

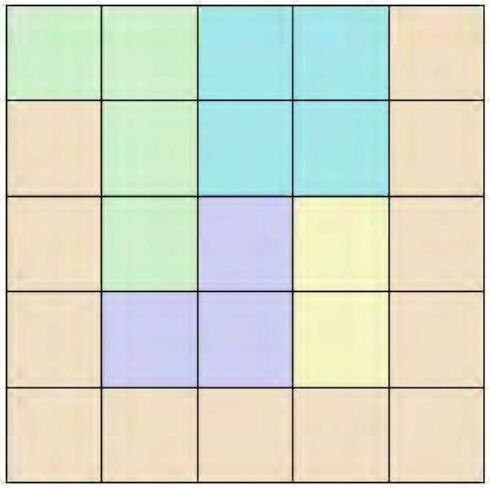
Applications

Formal Definition

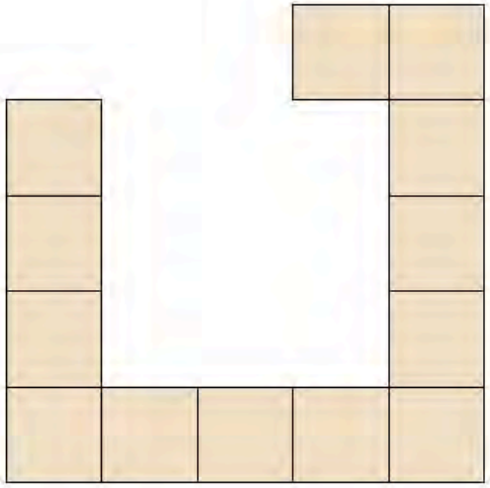
An assembly is interlocking if only one movable part (key), while all other parts, as well as **any subset** of the parts, are immobilized

↕

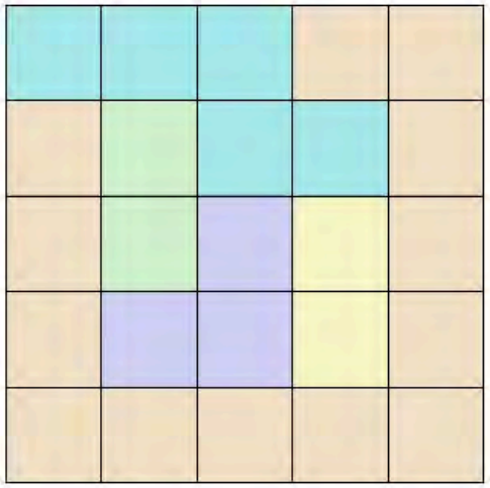
Test interlocking has exponential complexity!!!



Non-interlocking ☹

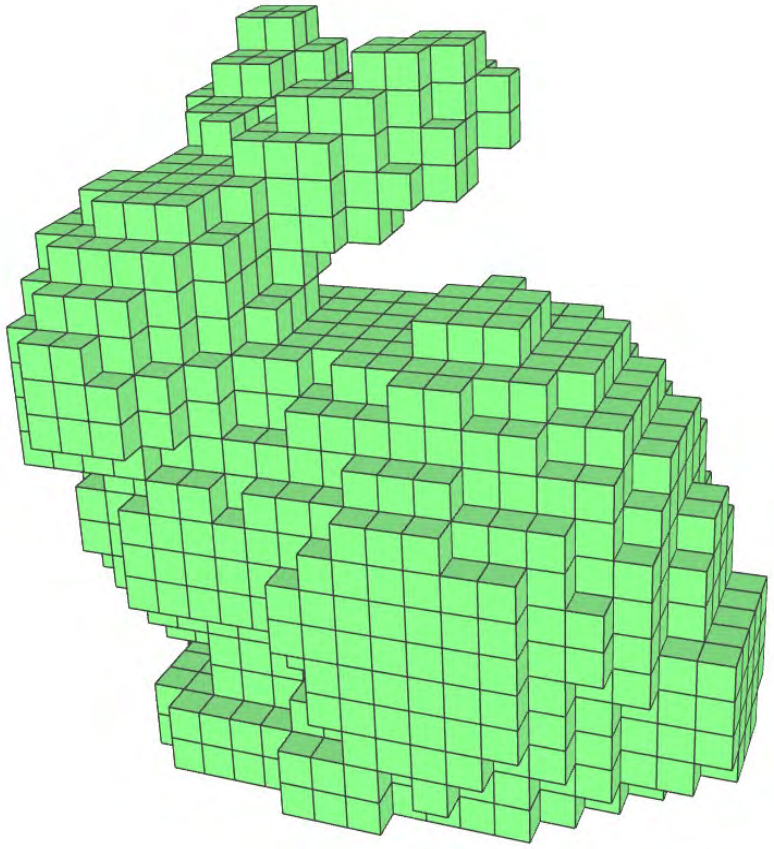


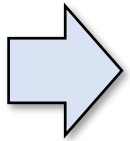
Interlocking 😀

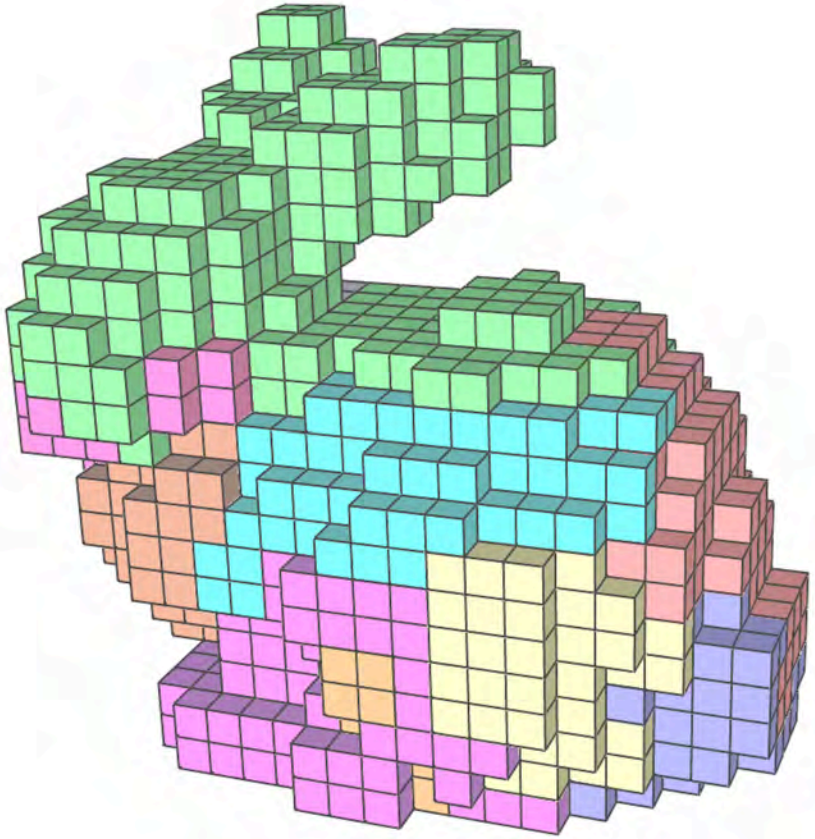


Deadlocking ☹

Interlocking Assembly: Design Problem



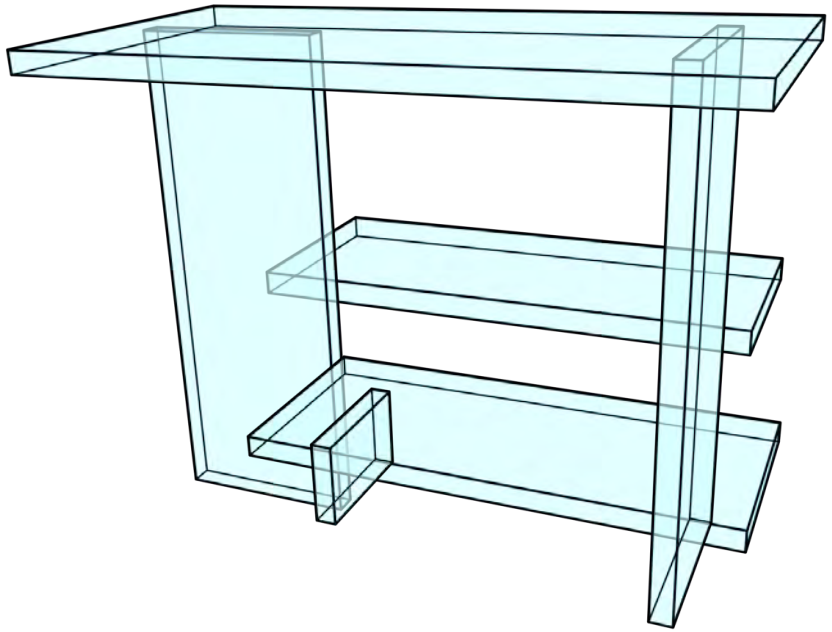




Input: target shape

Output: interlocking assembly

Interlocking Assembly: Design Problem

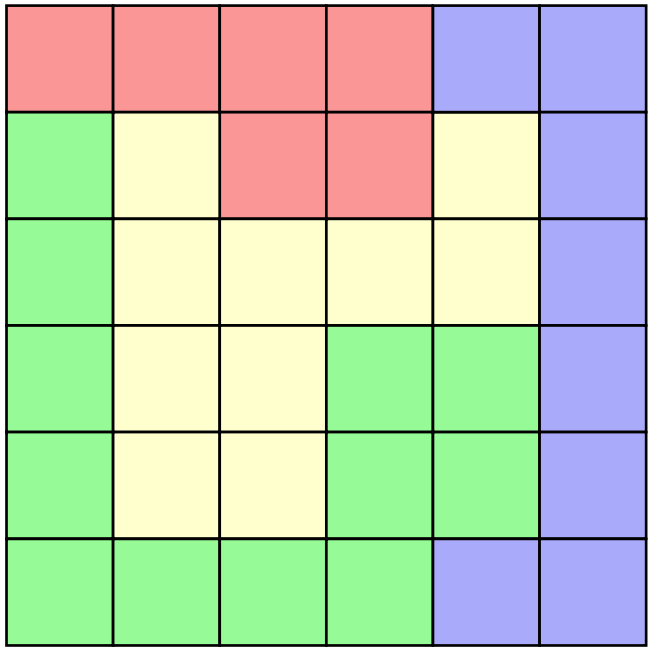


Input: target shape
(with segmented parts)

Output: interlocking assembly

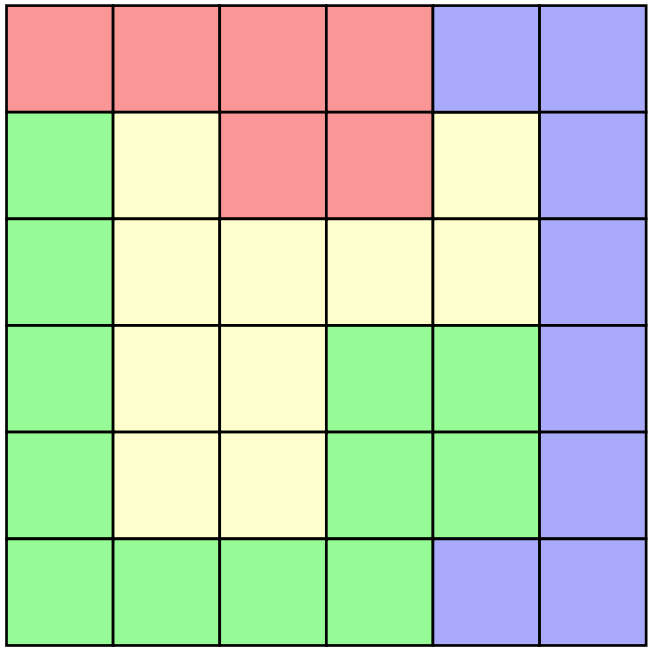
Challenge #1 Test Interlocking

- Need to test immobilization of every single part and every subset of parts



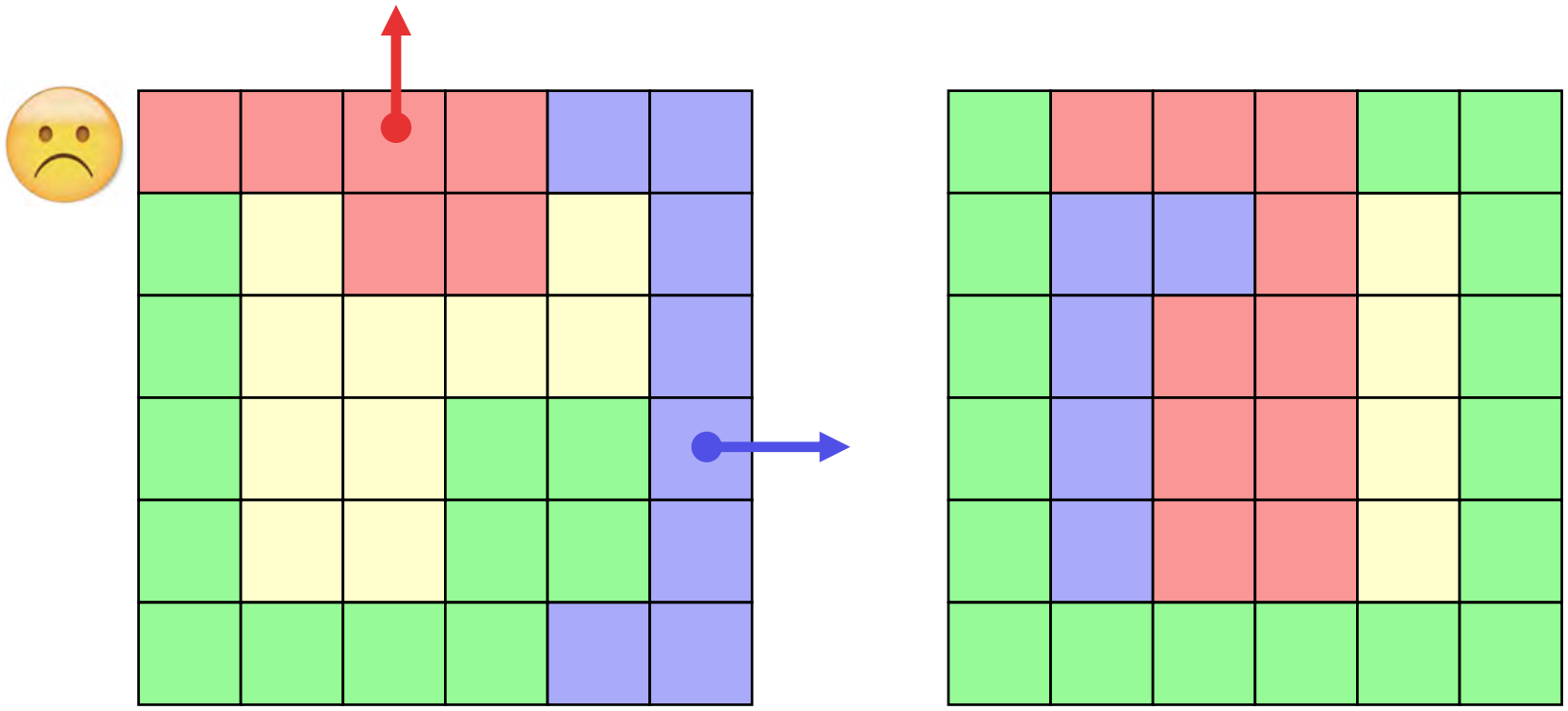
Challenge #1 Test Interlocking

- Need to test immobilization of every single part and every subset of parts



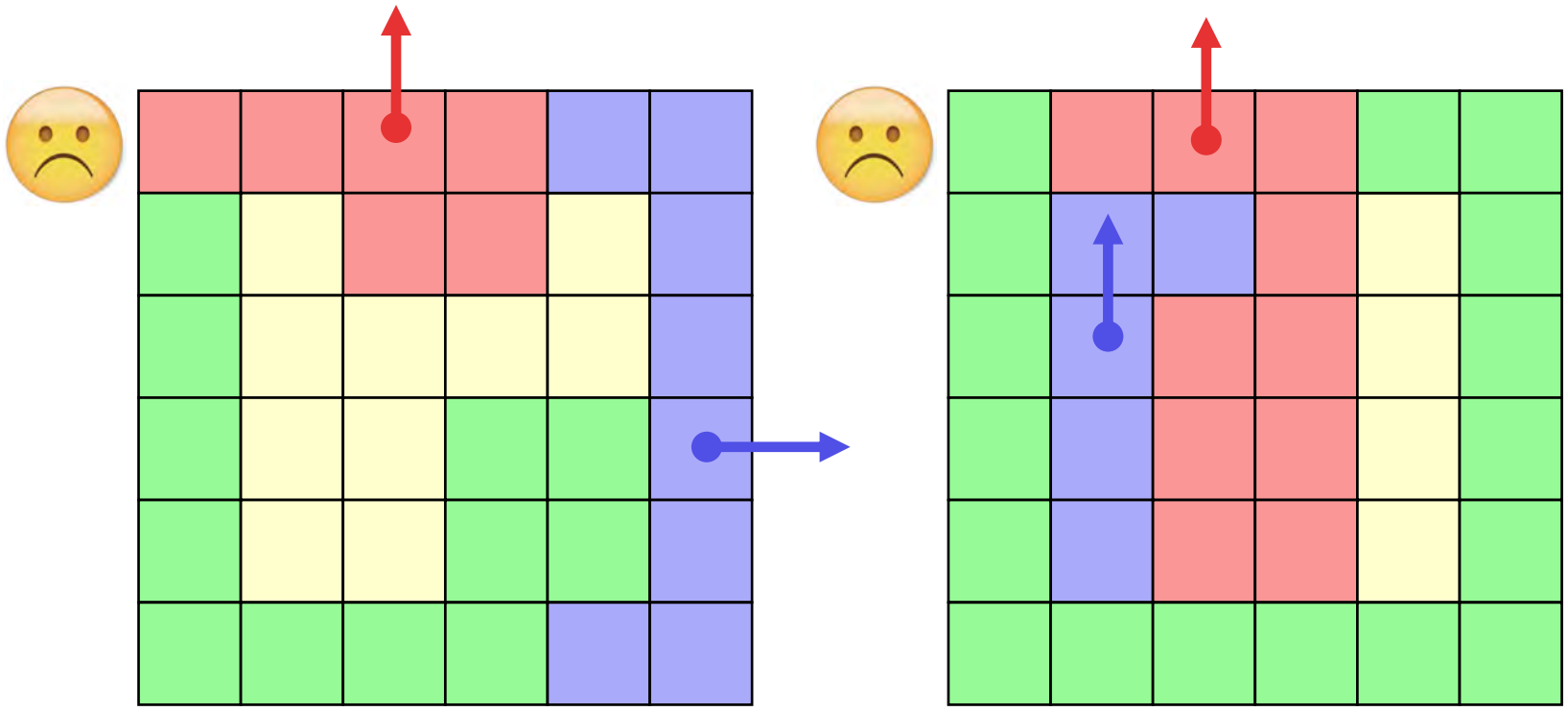
Challenge #1 Test Interlocking

- Need to test immobilization of every single part and every subset of parts



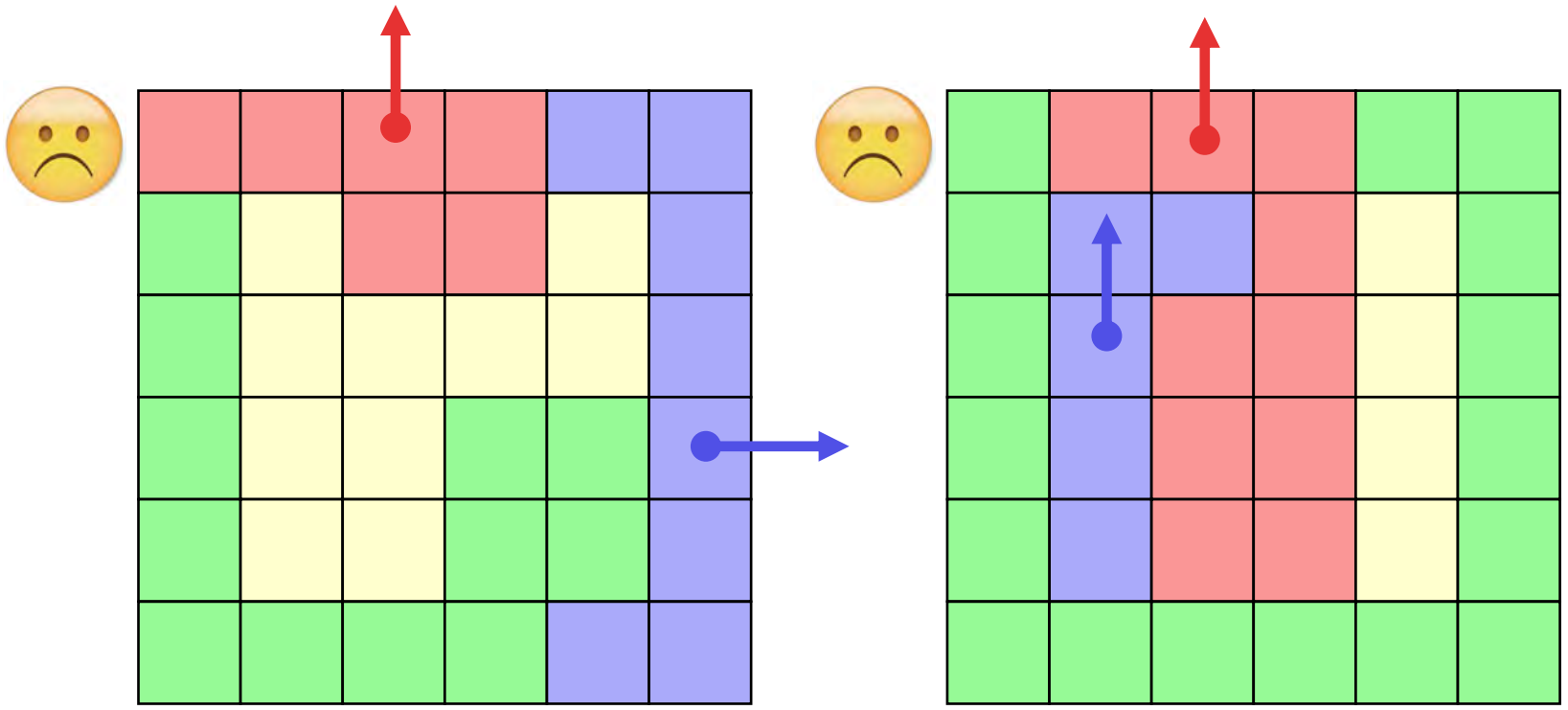
Challenge #1 Test Interlocking

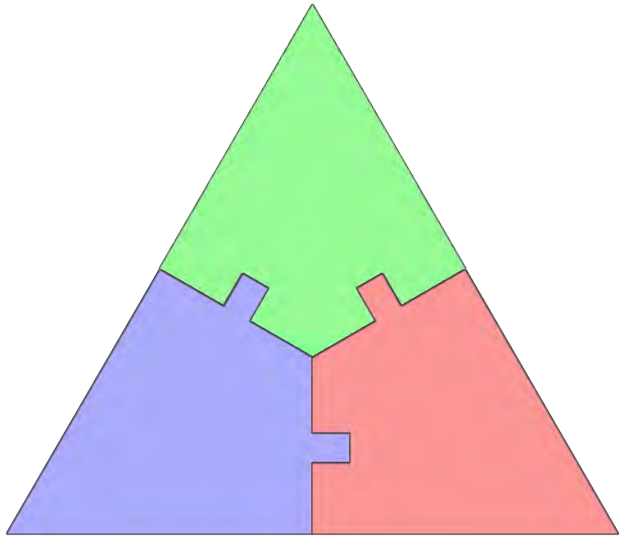
- Need to test immobilization of every single part and every subset of parts



Challenge #1 Test Interlocking

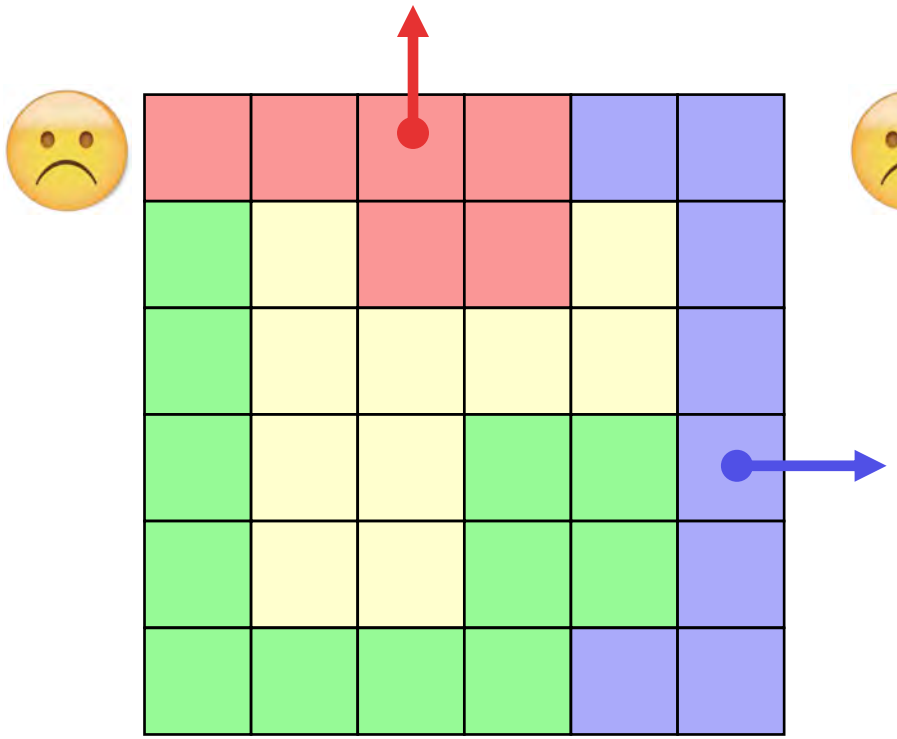
- Need to test immobilization of every single part and every subset of parts



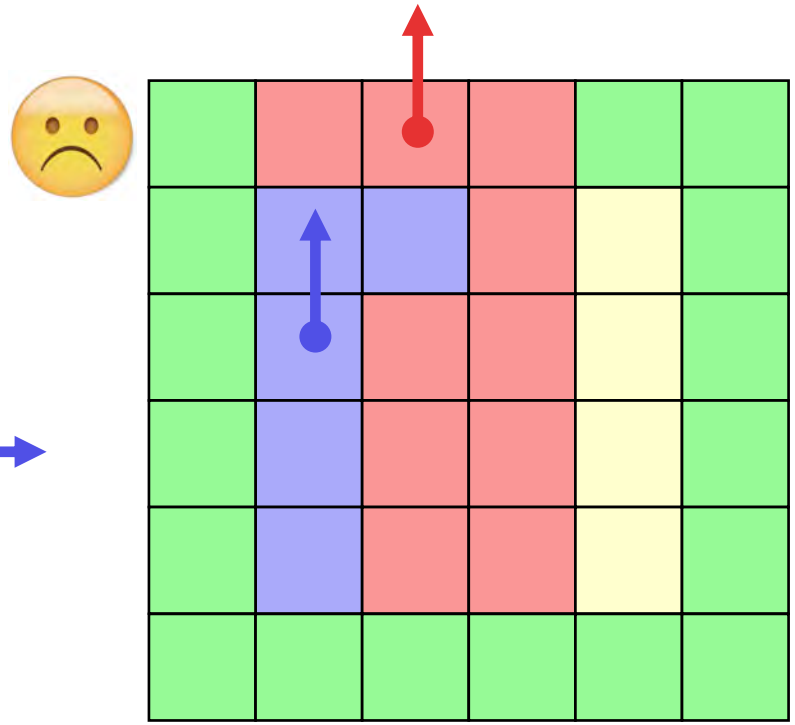


Challenge #1 Test Interlocking

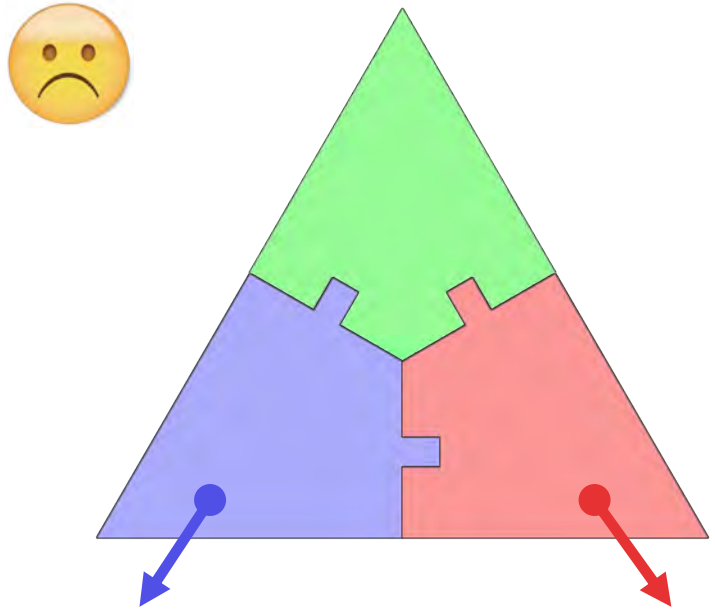
- Need to test immobilization of every single part and every subset of parts
- A subset of parts can be movable along different directions simultaneously



Single-direction movement



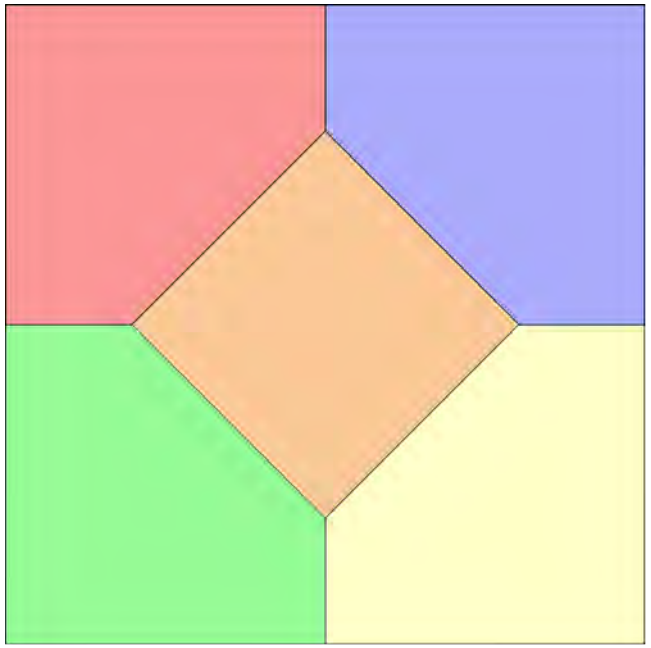
Single-direction movement



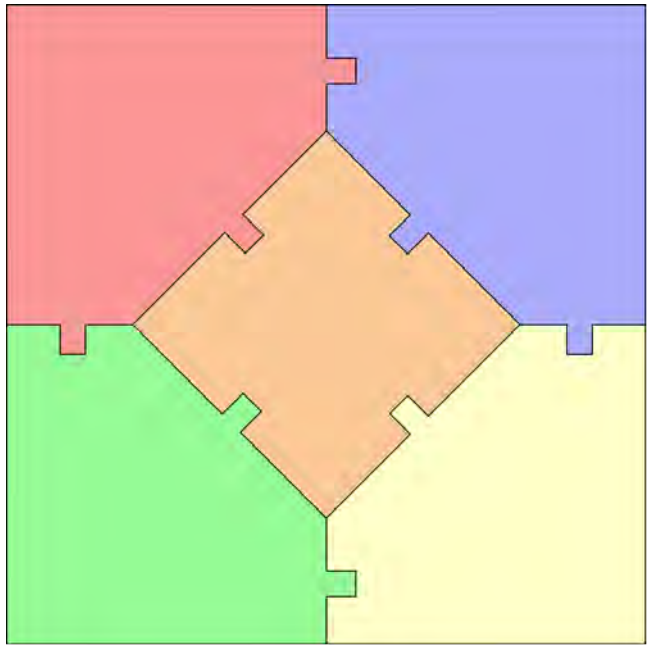
Multi-direction movement

Challenge #2 Disassemblable

Assembly can be progressively disassembled into a set of individual parts



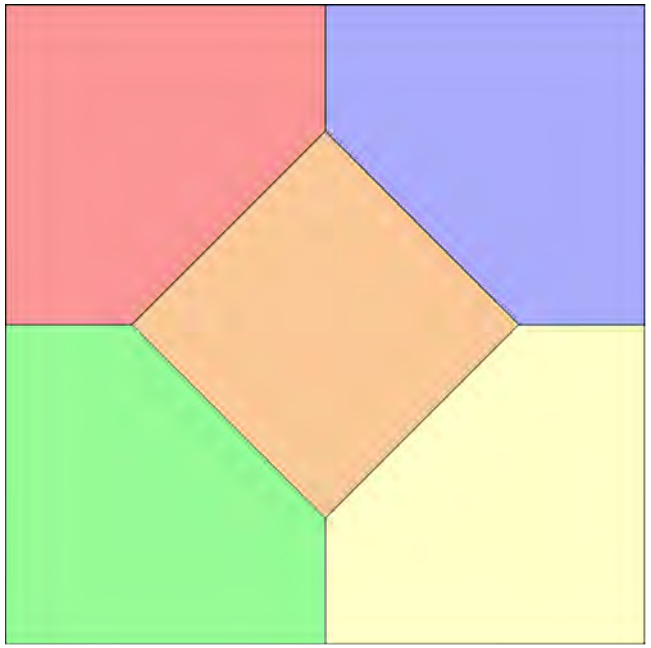
Input: segmented parts



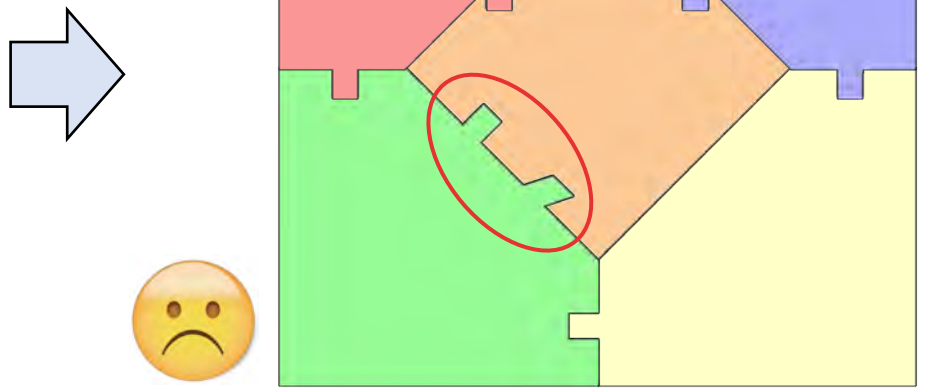
Deadlocking assembly

Challenge #2 Disassemblable

Assembly can be progressively disassembled into a set of individual parts



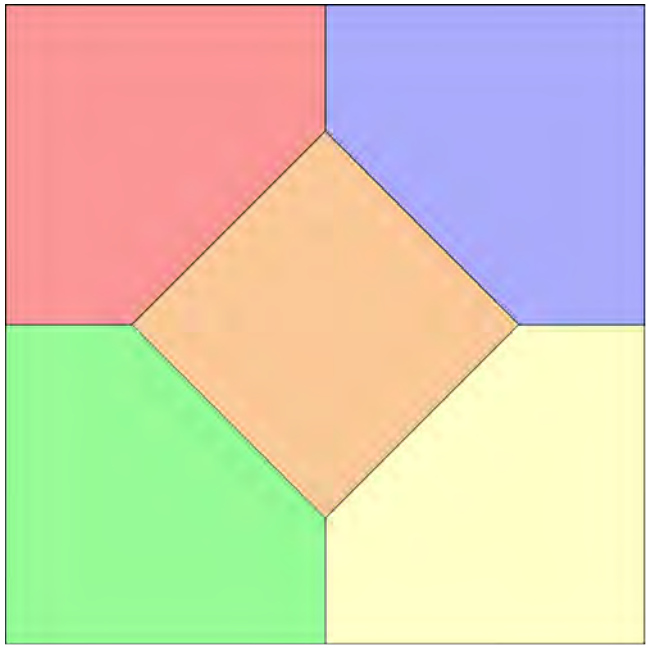
Input: segmented parts

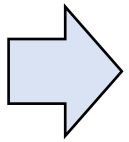


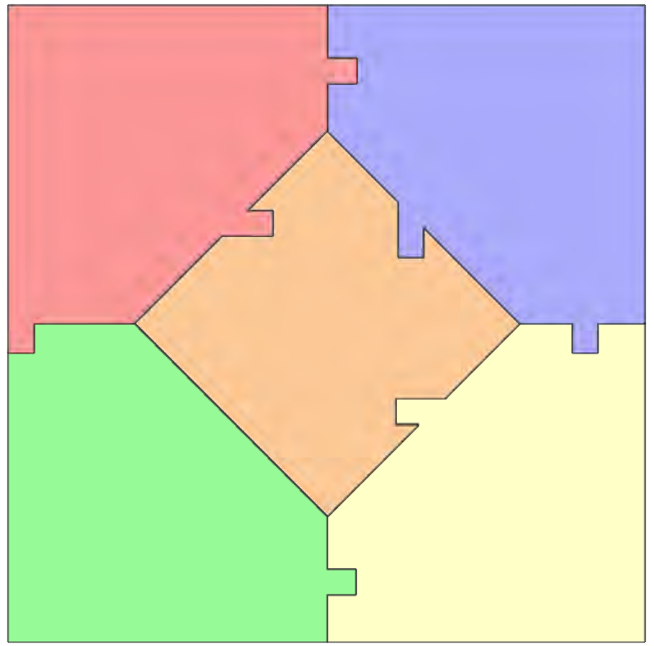
Deadlocking sub-assembly

Challenge #3 Parts Immobilization

Need to construct parts geometry such that every single part and every subset of parts are immobilized, except the key



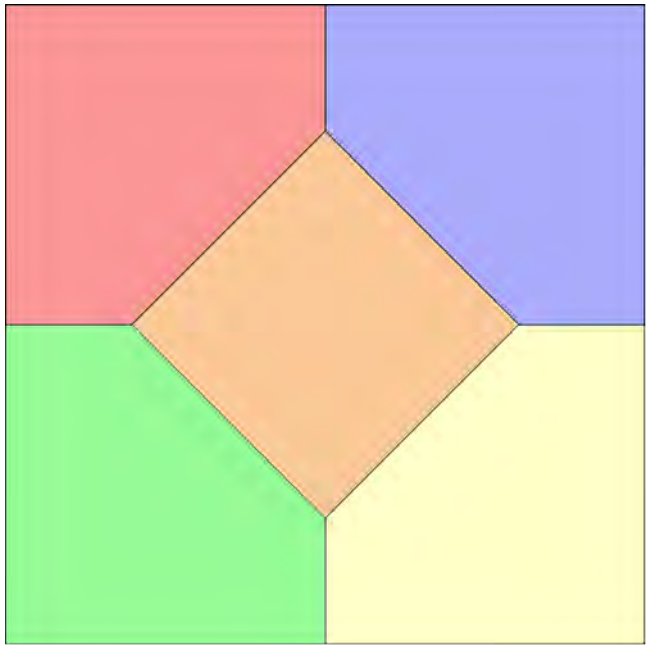




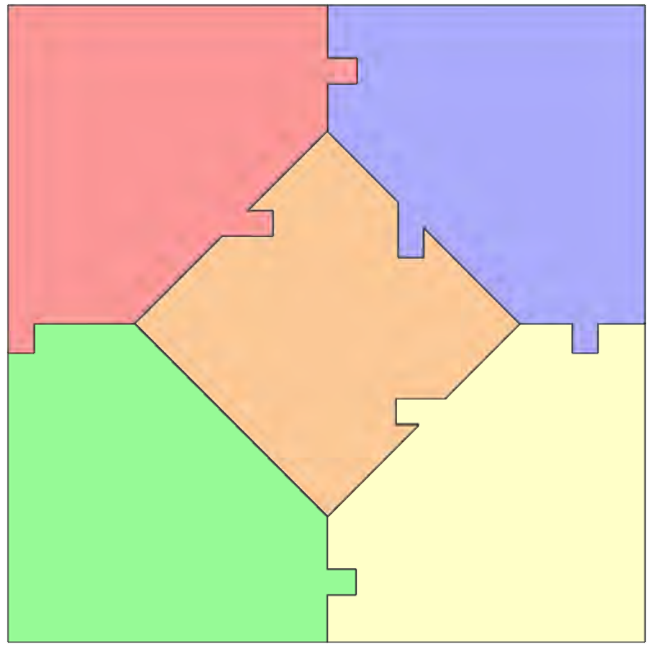
Input: segmented parts

Challenge #3 Parts Immobilization

Need to construct parts geometry such that every single part and every subset of parts are immobilized, except the key



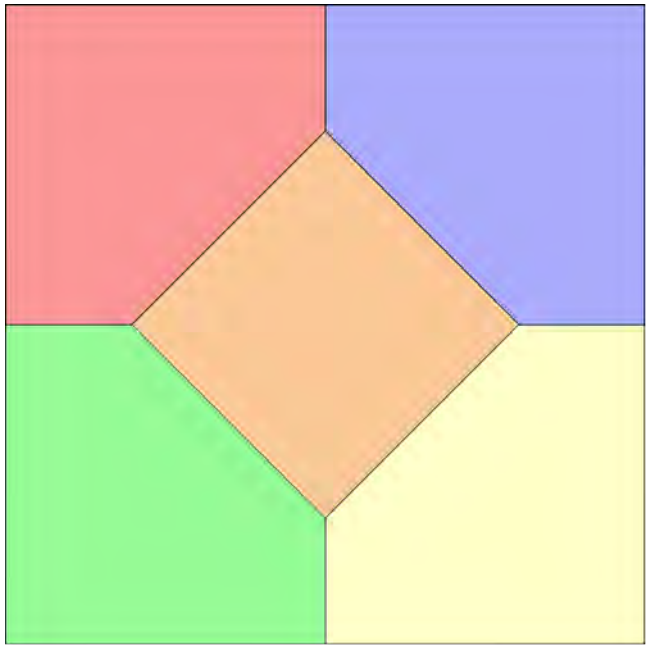
Input: segmented parts

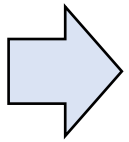


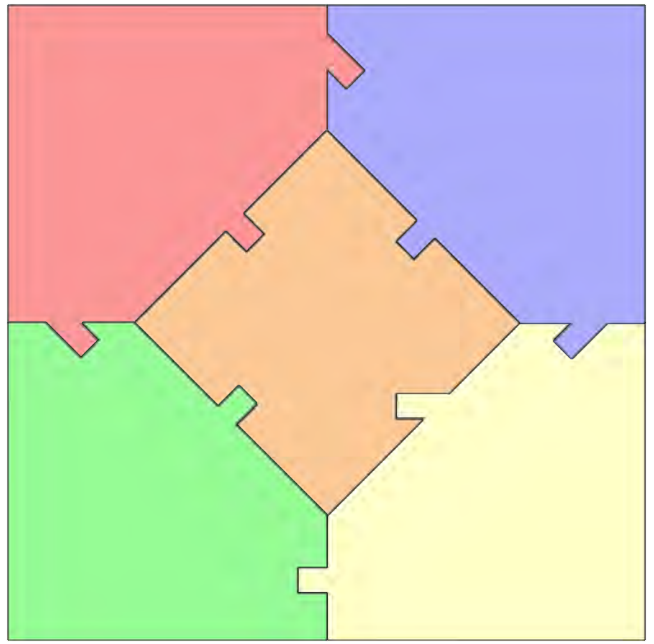
Movable parts group

Challenge #3 Parts Immobilization

Need to construct parts geometry such that every single part and every subset of parts are immobilized, except the key



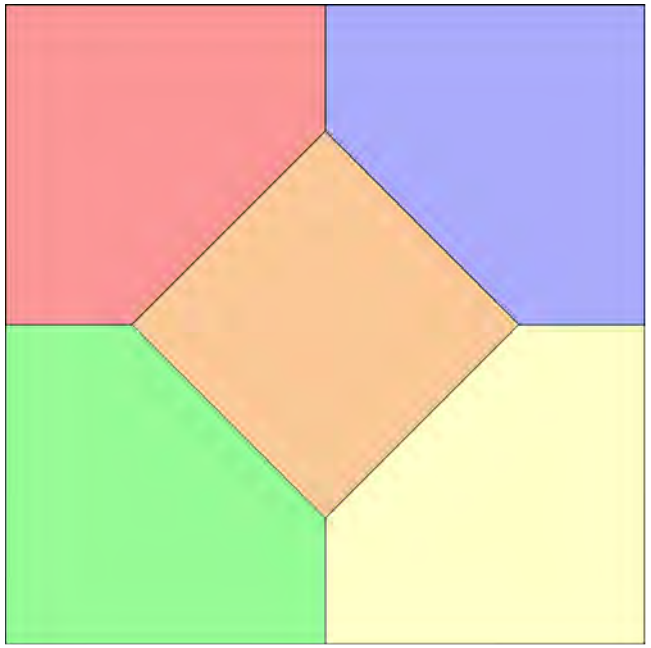




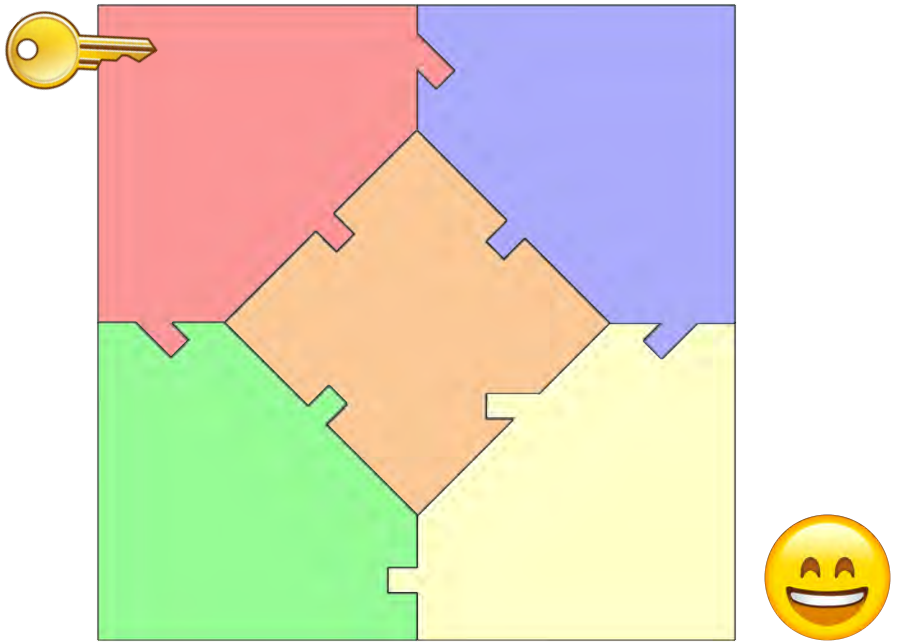
Input: segmented parts

Challenge #3 Parts Immobilization

Need to construct parts geometry such that every single part and every subset of parts are immobilized, except the key



Input: segmented parts



Interlocking Assembly

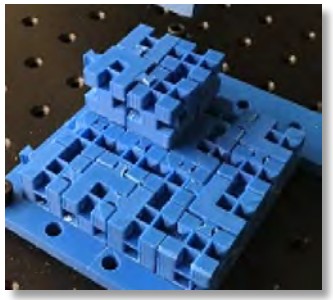
Interlocking Assembly: Related Works



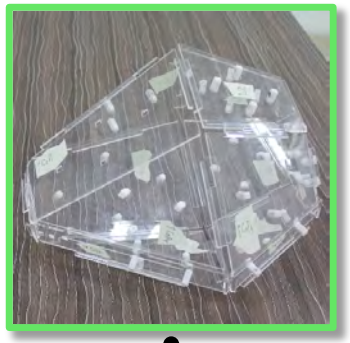



ancient 1978 2011 2012 2015 2016 2017 2018

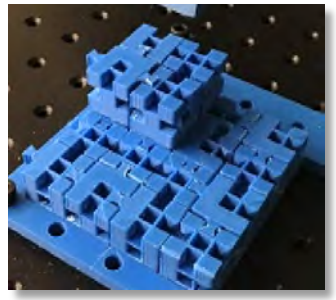


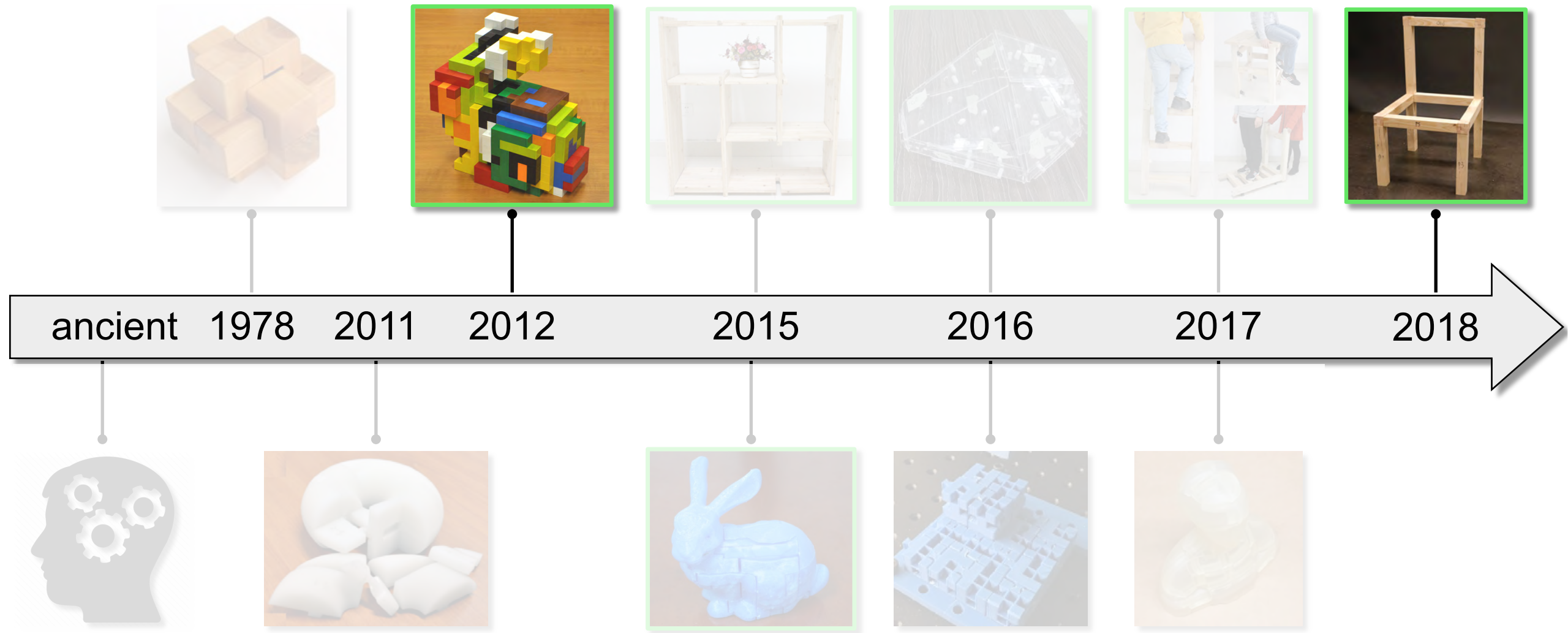

Interlocking Assembly: Related Works



ancient 1978 2011 2012 2015 2016 2017 2018



Interlocking Assembly: Related Works



Overview

Recursive Interlocking Puzzles

SIGGRAPH Asia 2012

DESIA: A General Framework for Designing Interlocking Assemblies

SIGGRAPH Asia 2018

Overview

Recursive Interlocking Puzzles

SIGGRAPH Asia 2012

DESIA: A General Framework for Designing Interlocking Assemblies

SIGGRAPH Asia 2018

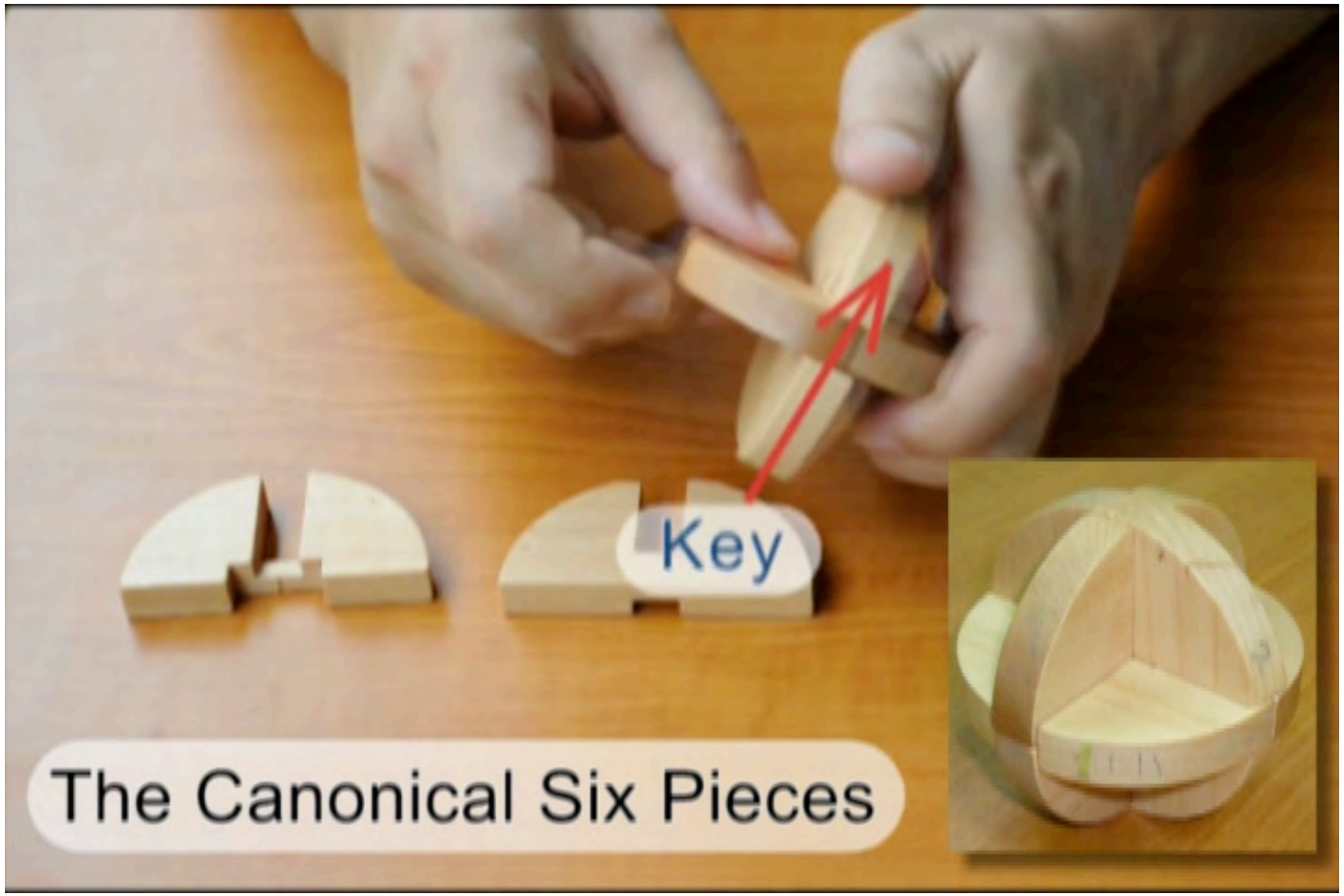
Interlocking Puzzles

2D puzzle

3D puzzle

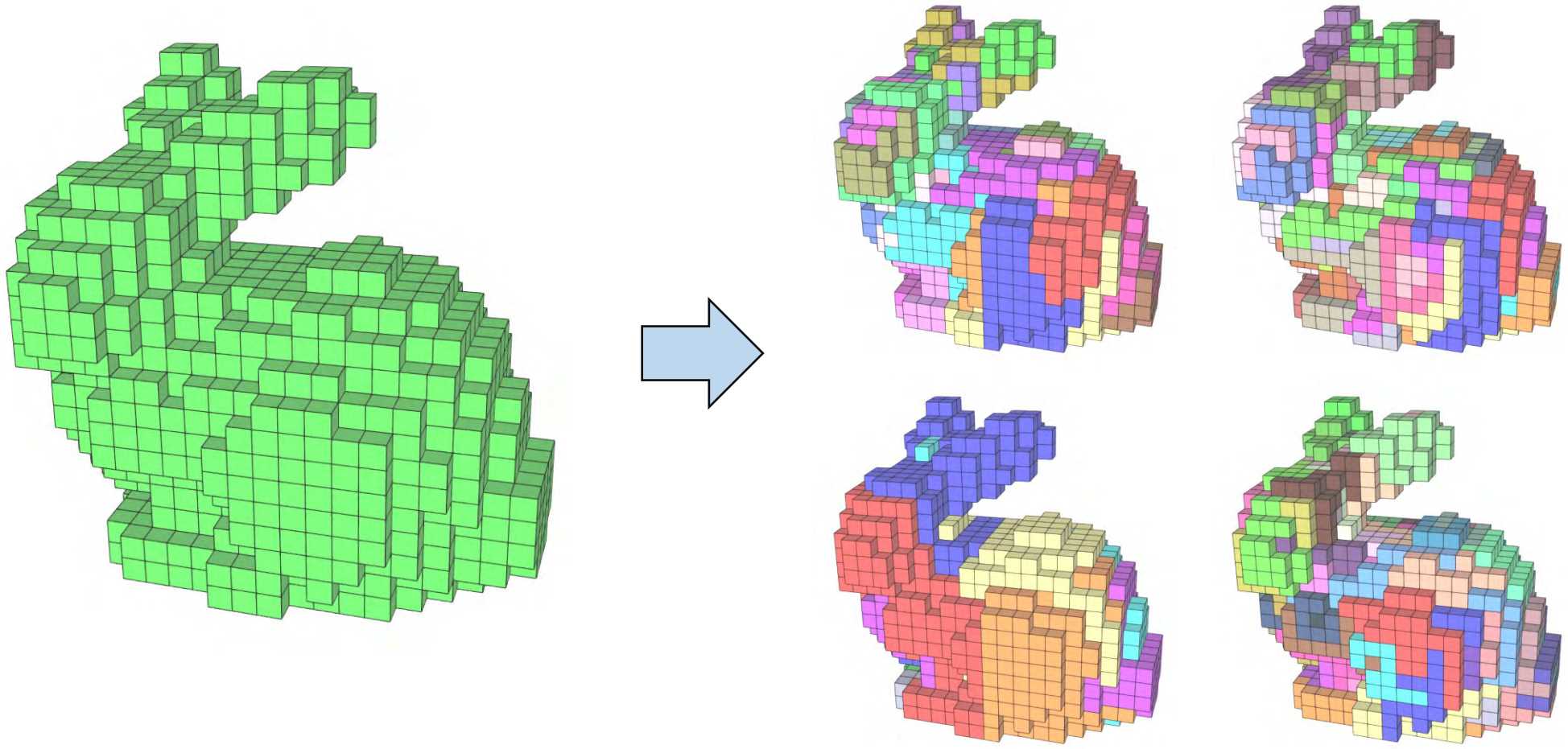
3D interlocking puzzle

Interlocking Puzzles



3D interlocking puzzle

Our Goal: Design Interlocking Puzzles

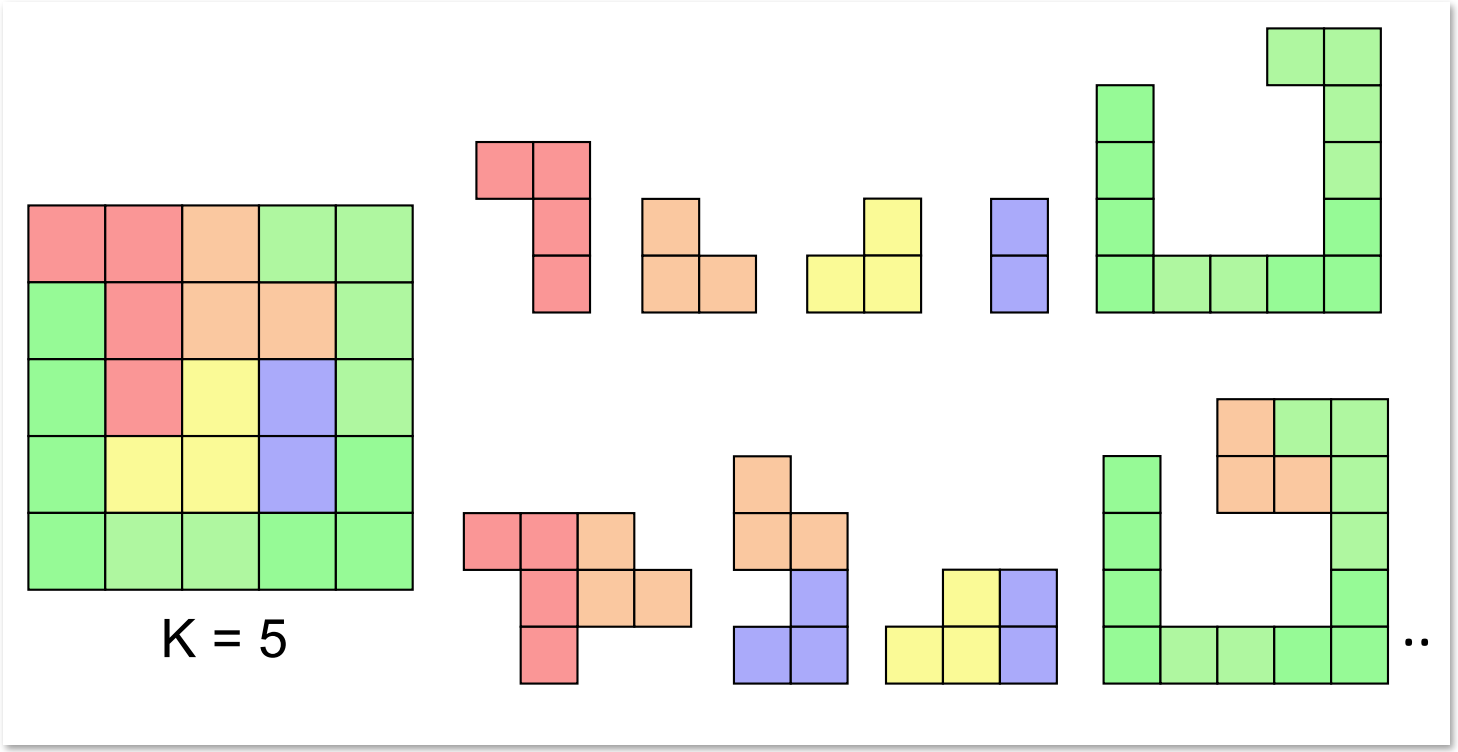


Input: voxelized model

Output: K interlocking parts

Test Interlocking: Parts Immobilization

An assembly is interlocking if only one movable part (key), while all other parts, as well as any subset of parts, are immobilized



C(K, 1)

+

C(K, 2)

...

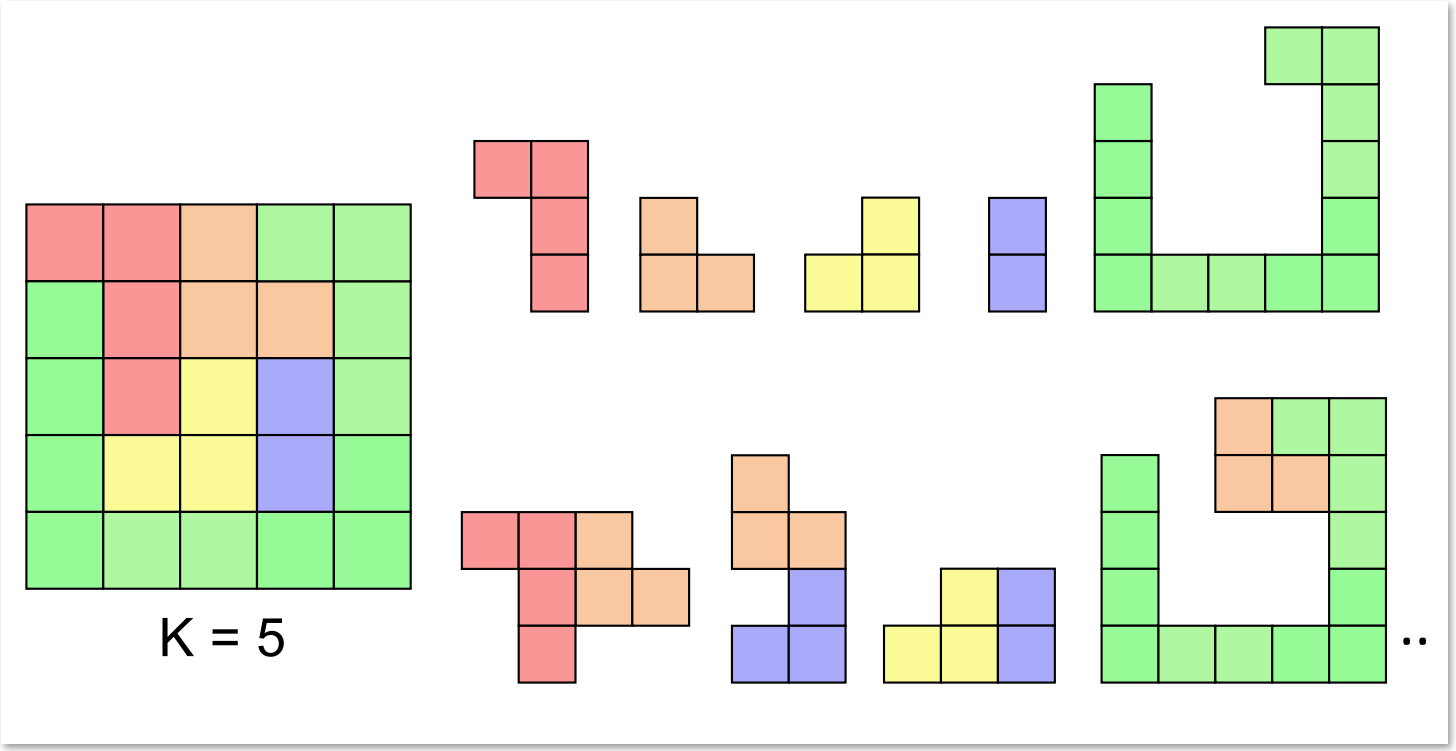
+

C(K, $\lfloor K/2 \rfloor$)

$= 2^{(K-1)}$ subsets of parts

Test Interlocking: Parts Immobilization

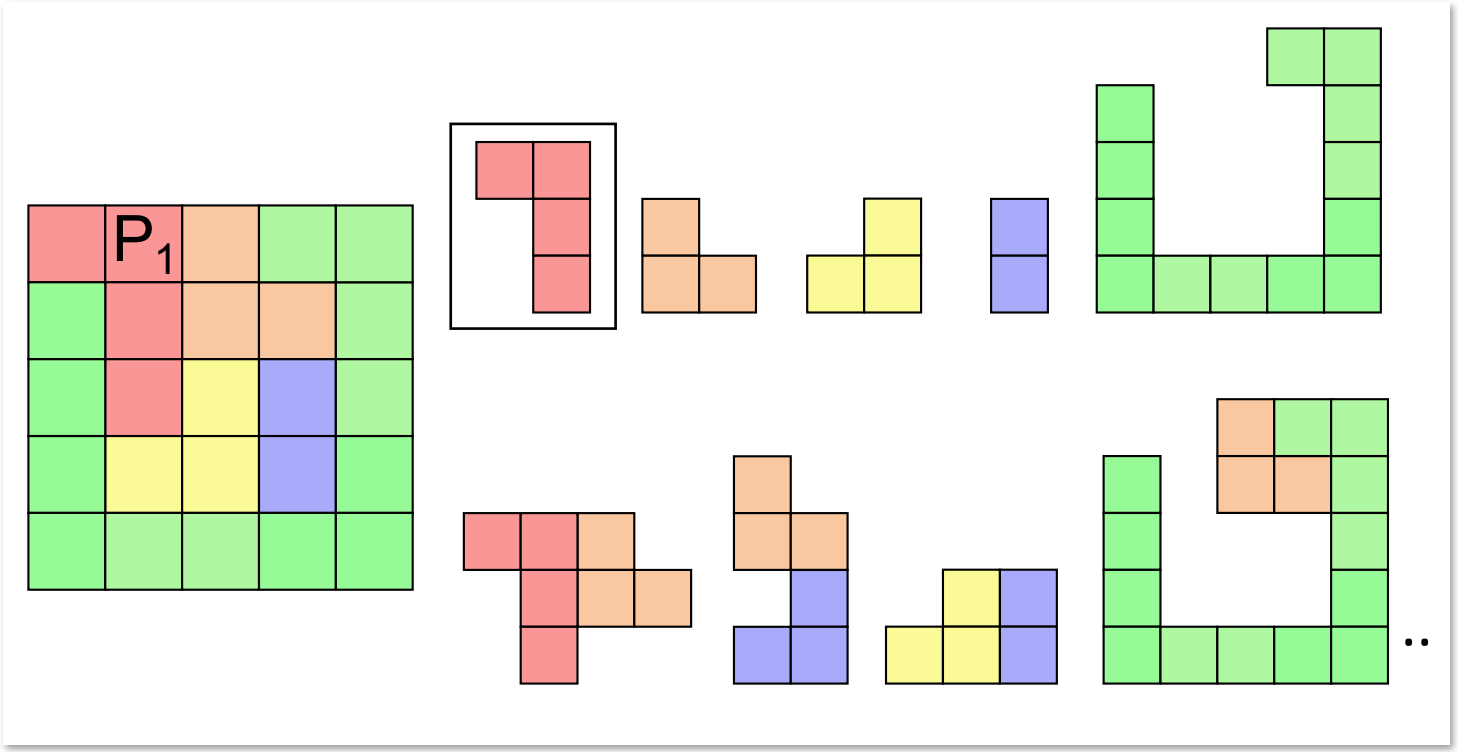
An assembly is interlocking if only one movable part (key), while all other parts, as well as any subset of parts, are immobilized



$2^{(K-1)}$ subsets of parts

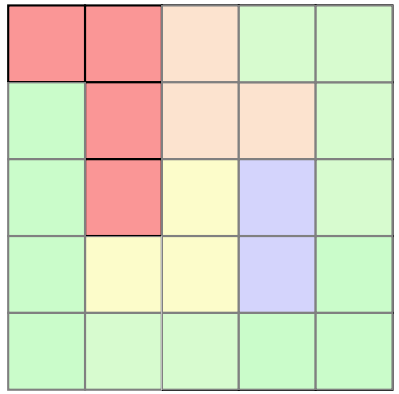
Test Interlocking: Parts Immobilization

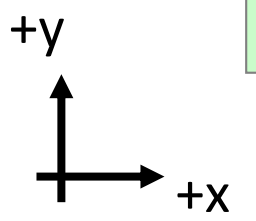
An assembly is interlocking if only one movable part (key), while all other parts, as well as any subset of parts, are immobilized



$2^{(K-1)}$ subsets of parts

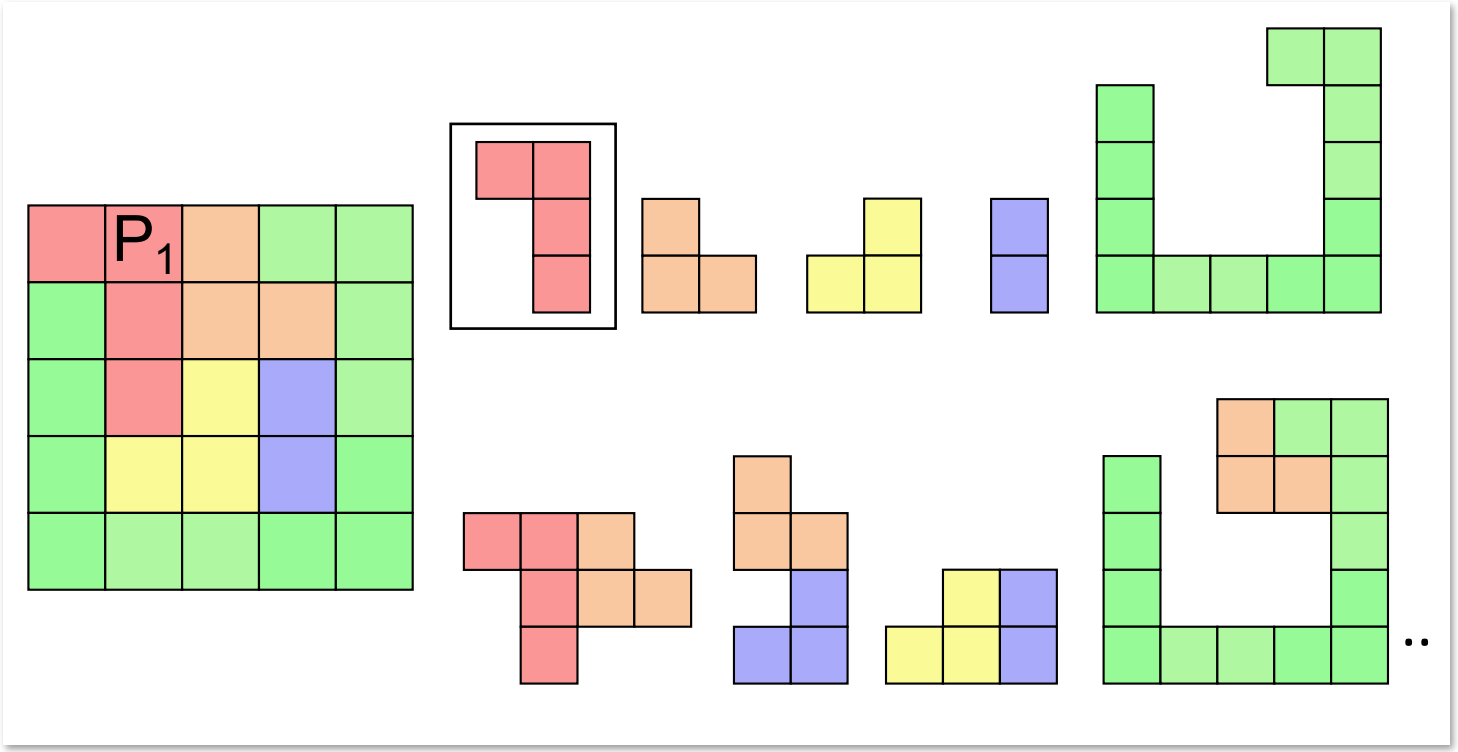
Test immobilization: P_1
(movable along –x, +x, -y, +y?)





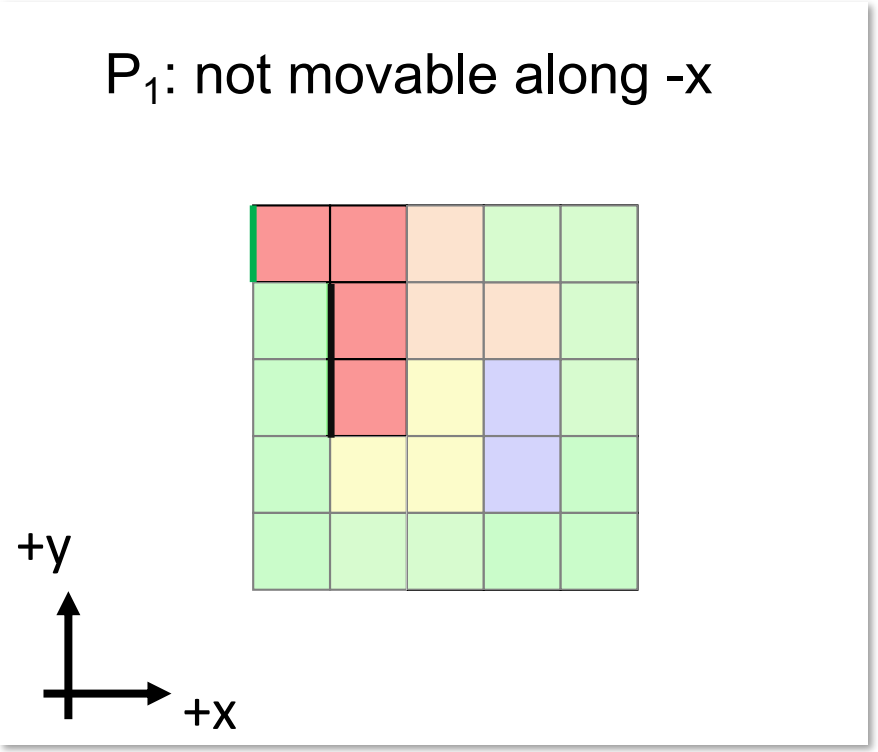
Test Interlocking: Parts Immobilization

An assembly is interlocking if only one movable part (key), while all other parts, as well as any subset of parts, are immobilized



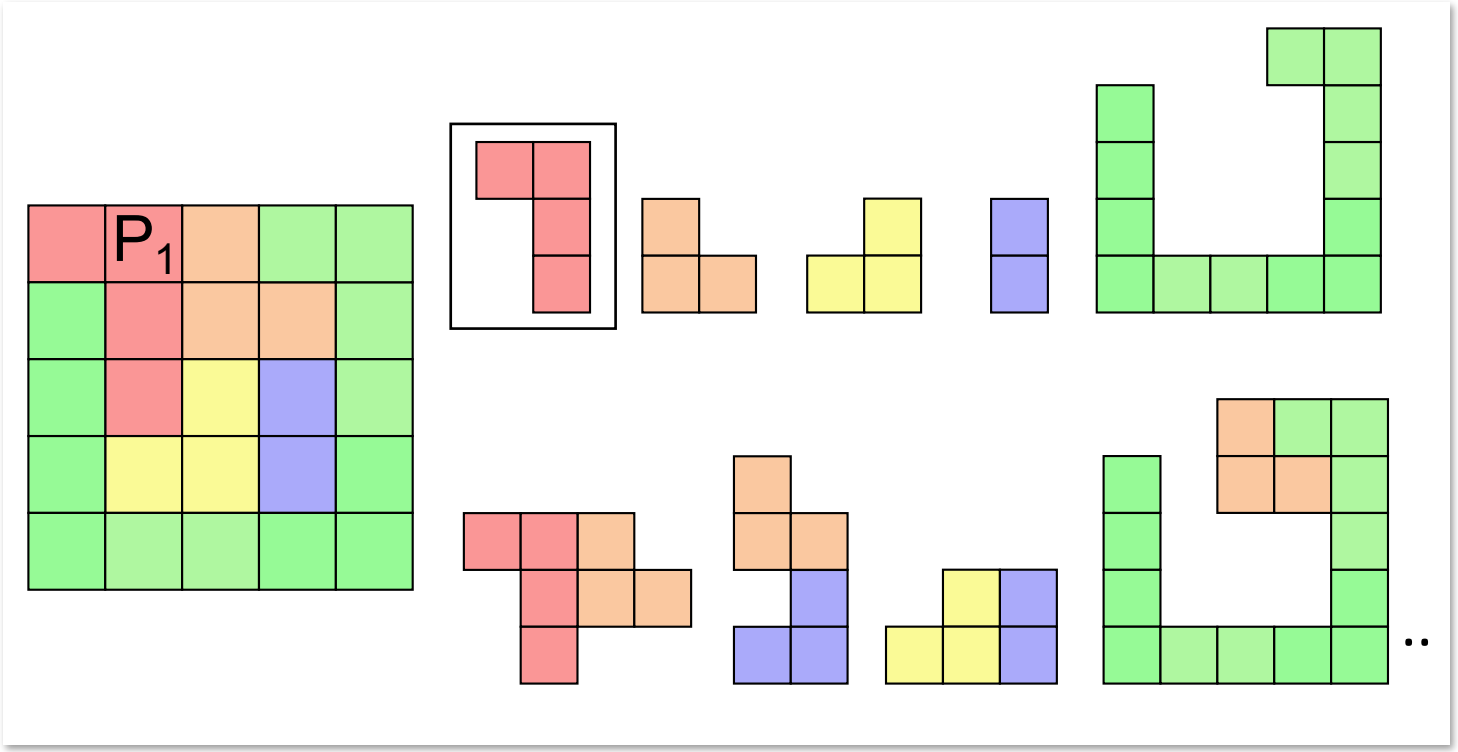
$2^{(K-1)}$ subsets of parts

P_1: not movable along -x



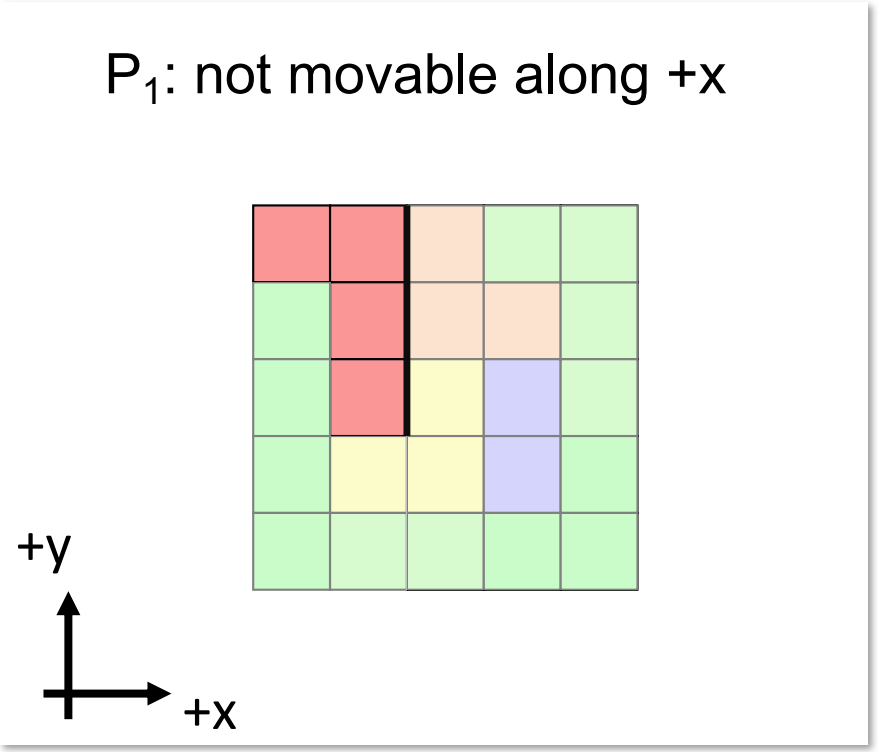
Test Interlocking: Parts Immobilization

An assembly is interlocking if only one movable part (key), while all other parts, as well as any subset of parts, are immobilized



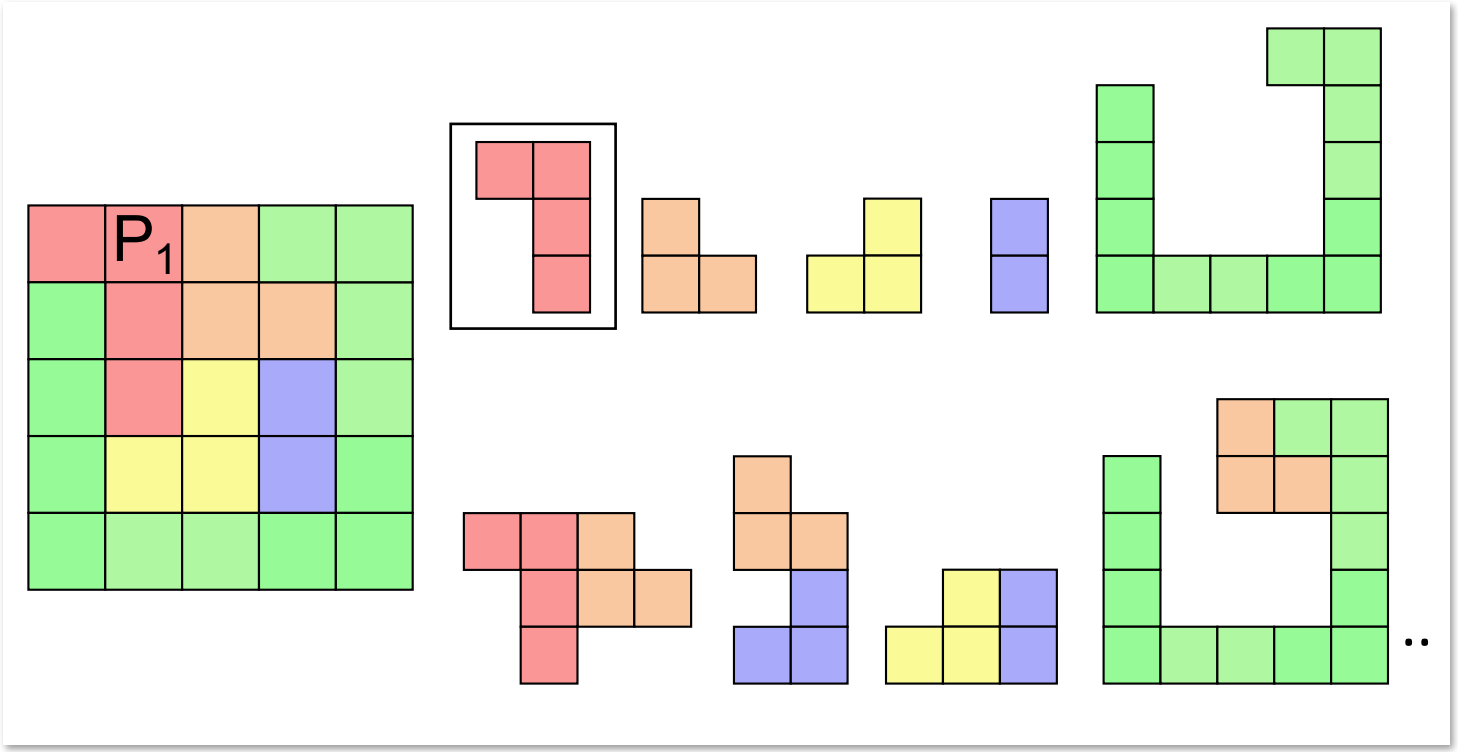
$2^{(K-1)}$ subsets of parts

P_1: not movable along +x



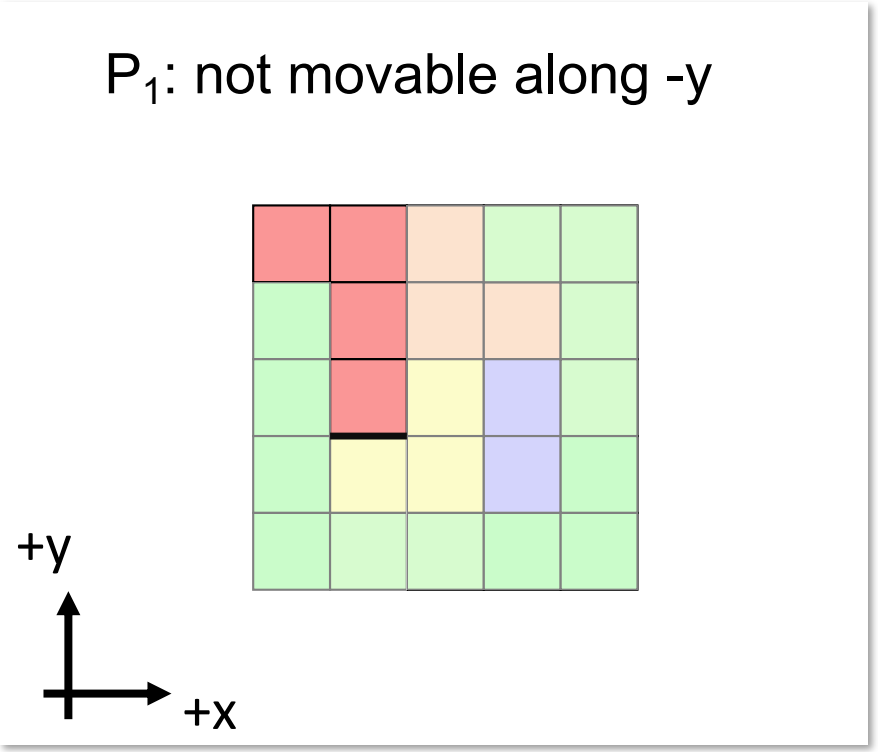
Test Interlocking: Parts Immobilization

An assembly is interlocking if only one movable part (key), while all other parts, as well as any subset of parts, are immobilized



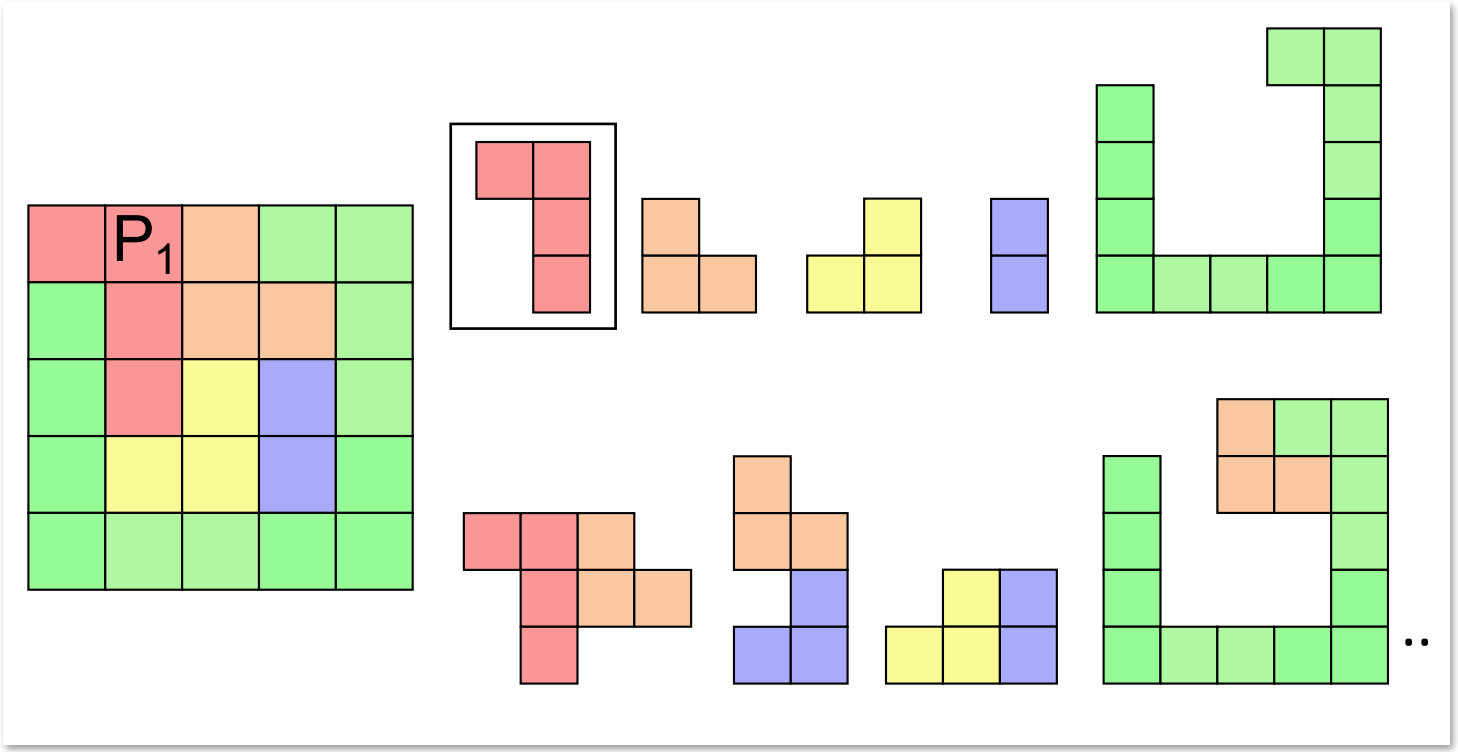
$2^{(K-1)}$ subsets of parts

P_1: not movable along -y

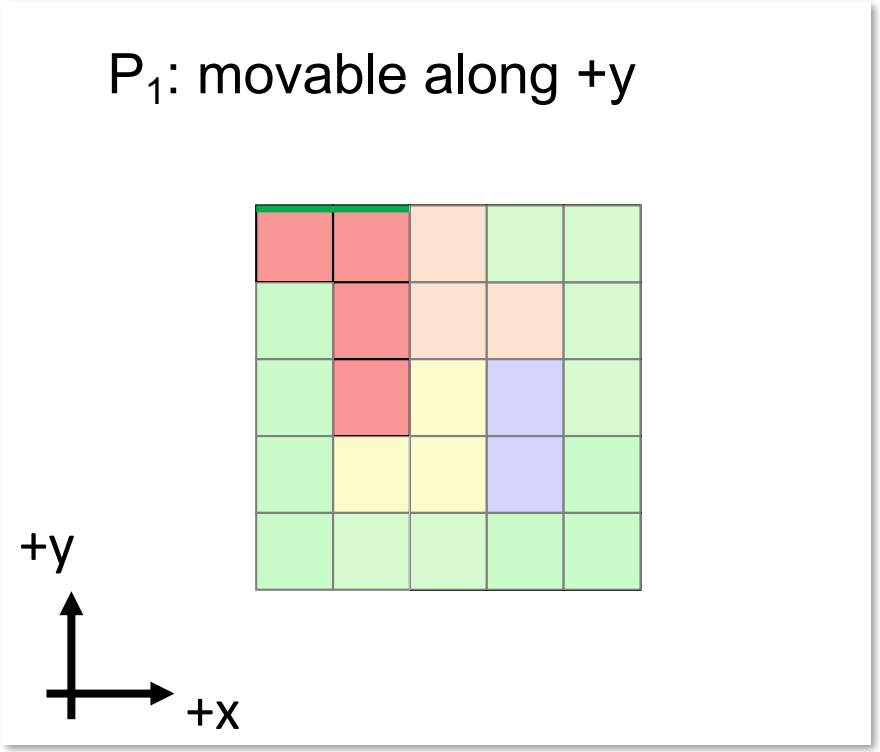


Test Interlocking: Parts Immobilization

An assembly is interlocking if only one movable part (key), while all other parts, as well as any subset of parts, are immobilized

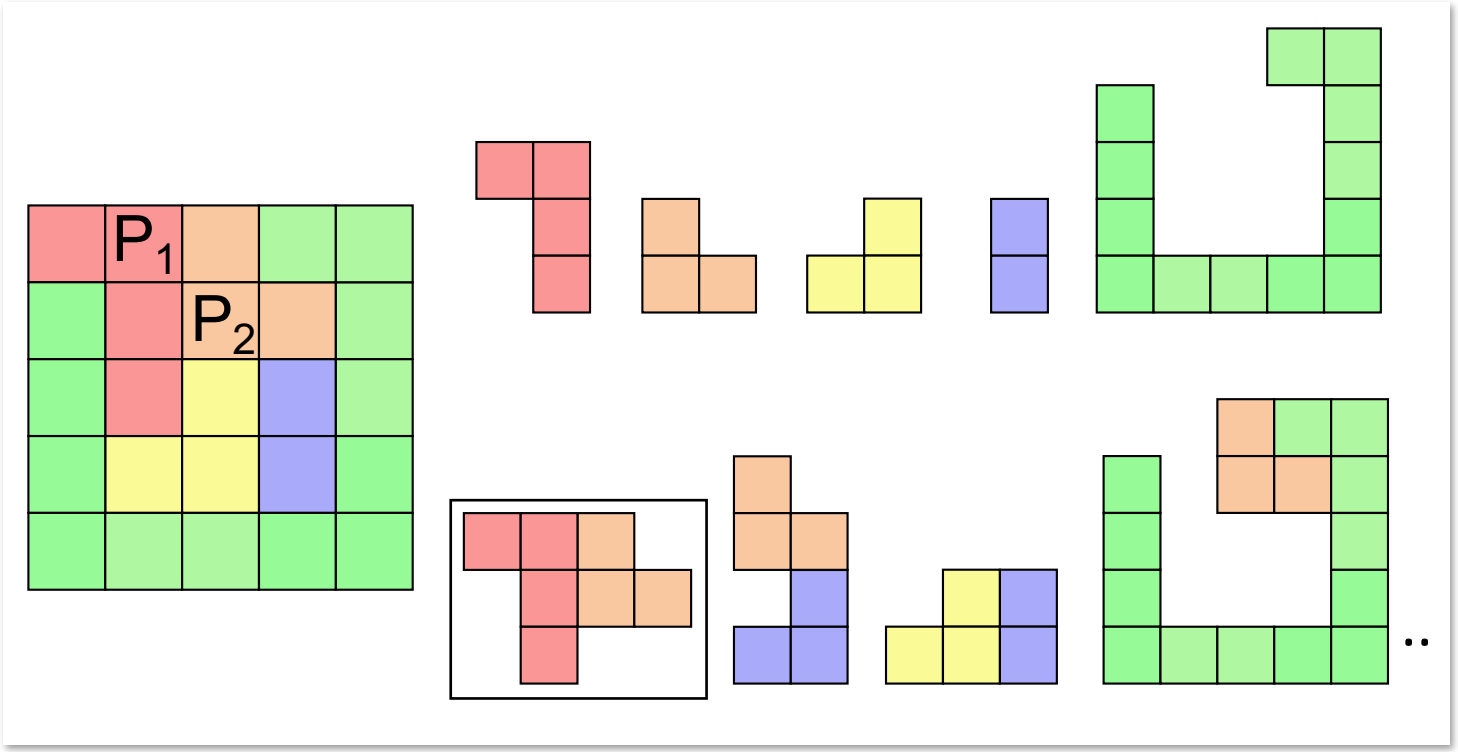


$2^{(K-1)}$ subsets of parts



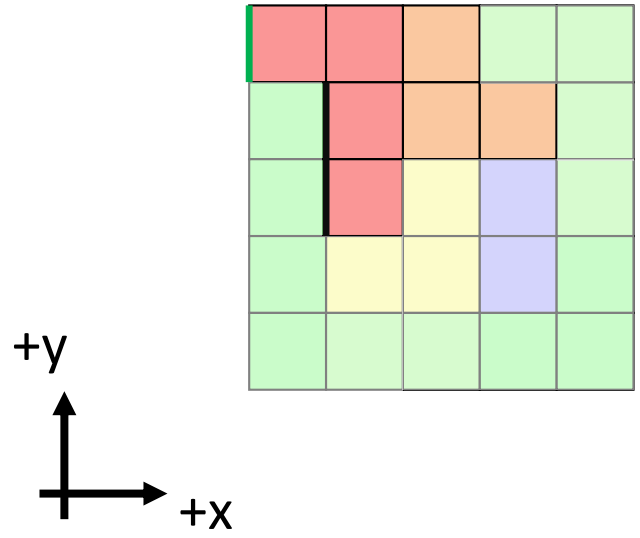
Test Interlocking: Parts Immobilization

An assembly is interlocking if only one movable part (key), while all other parts, as well as any subset of parts, are immobilized



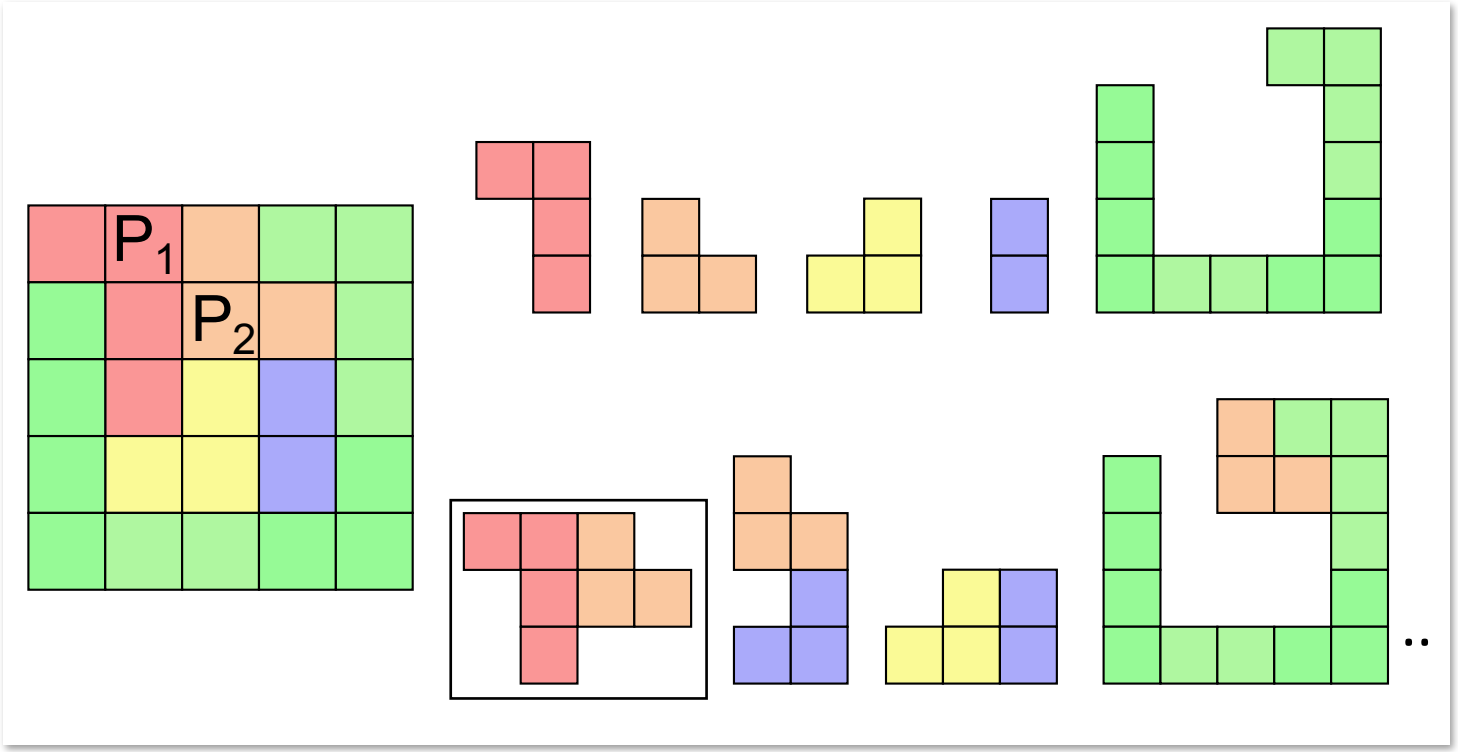
$2^{(K-1)}$ subsets of parts

P_1P_2 : not movable along -x



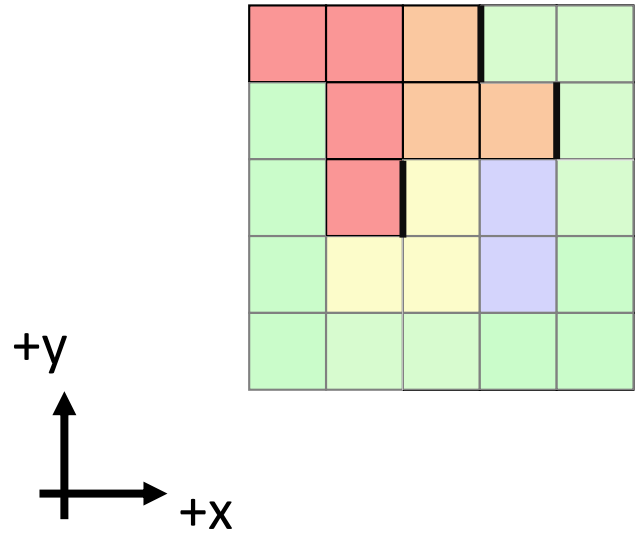
Test Interlocking: Parts Immobilization

An assembly is interlocking if only one movable part (key), while all other parts, as well as any subset of parts, are immobilized



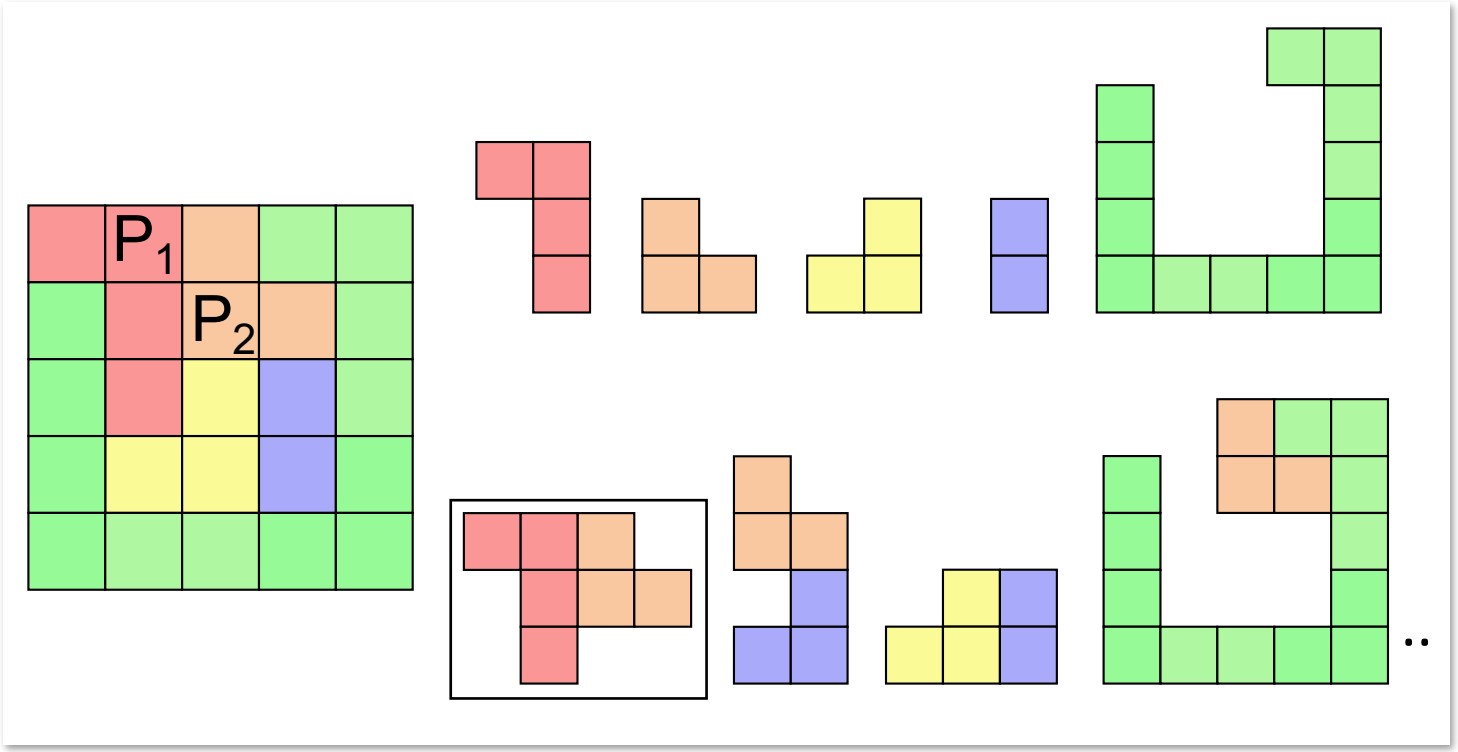
$2^{(K-1)}$ subsets of parts

P_1P_2 : not movable along +x



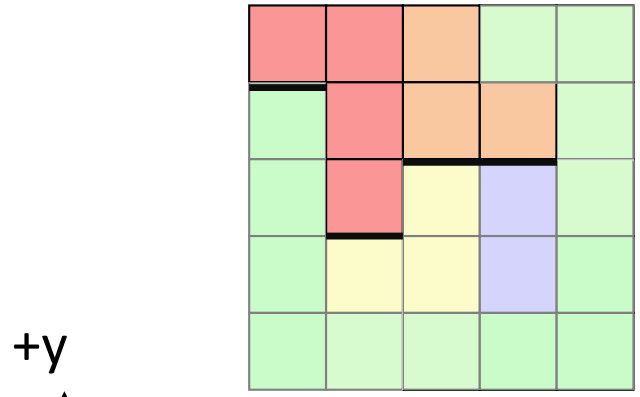
Test Interlocking: Parts Immobilization

An assembly is interlocking if only one movable part (key), while all other parts, as well as any subset of parts, are immobilized



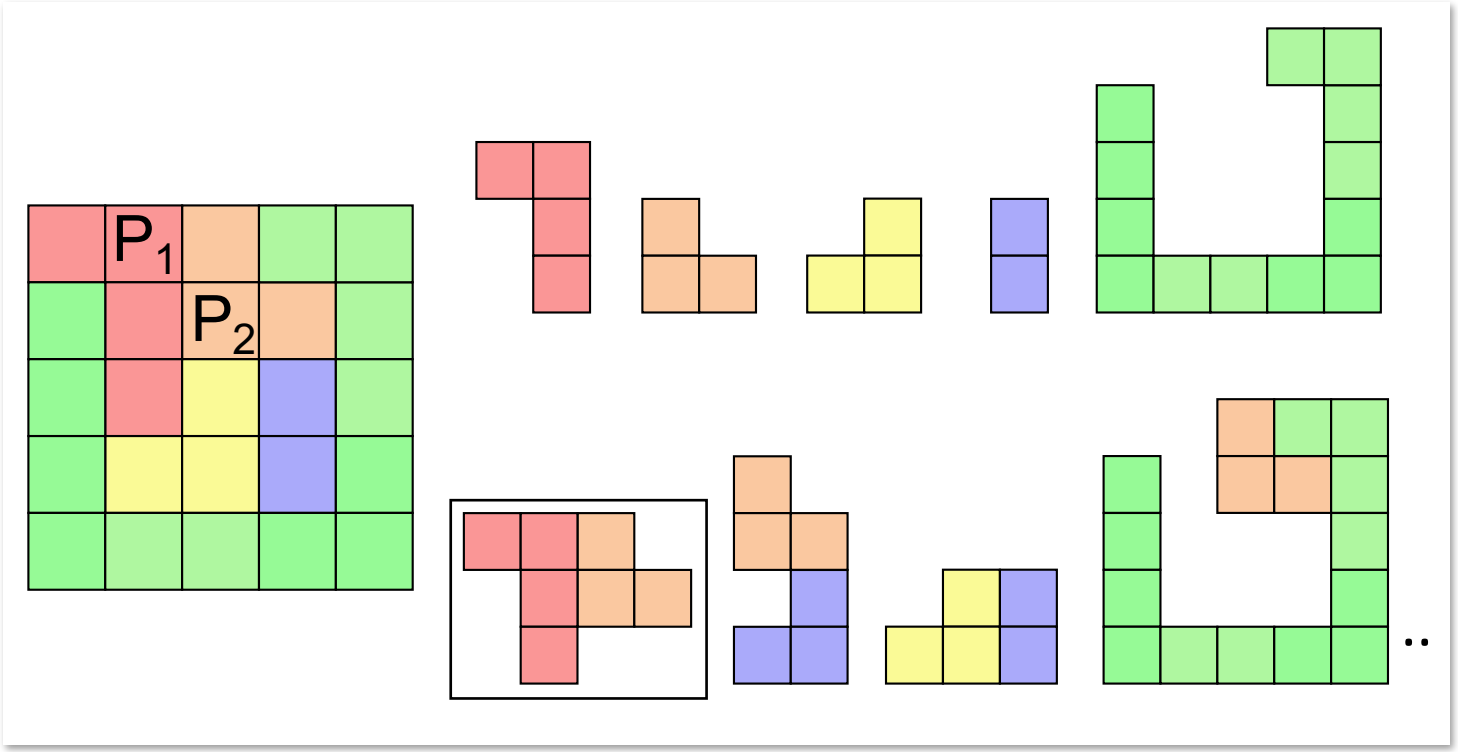
$2^{(K-1)}$ subsets of parts

P_1P_2 : not movable along -y

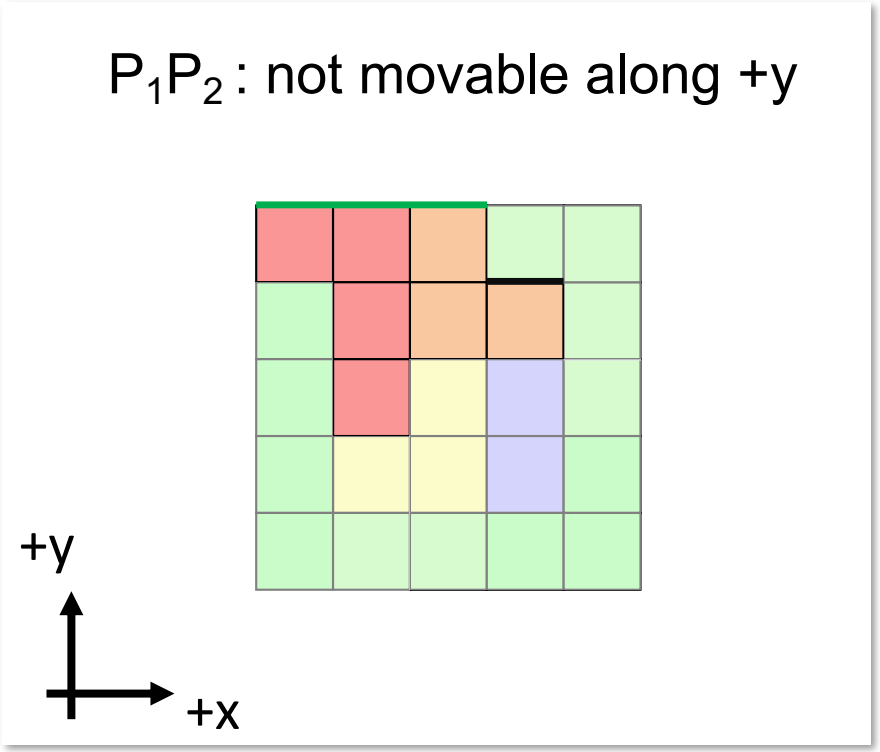


Test Interlocking: Parts Immobilization

An assembly is interlocking if only one movable part (key), while all other parts, as well as any subset of parts, are immobilized



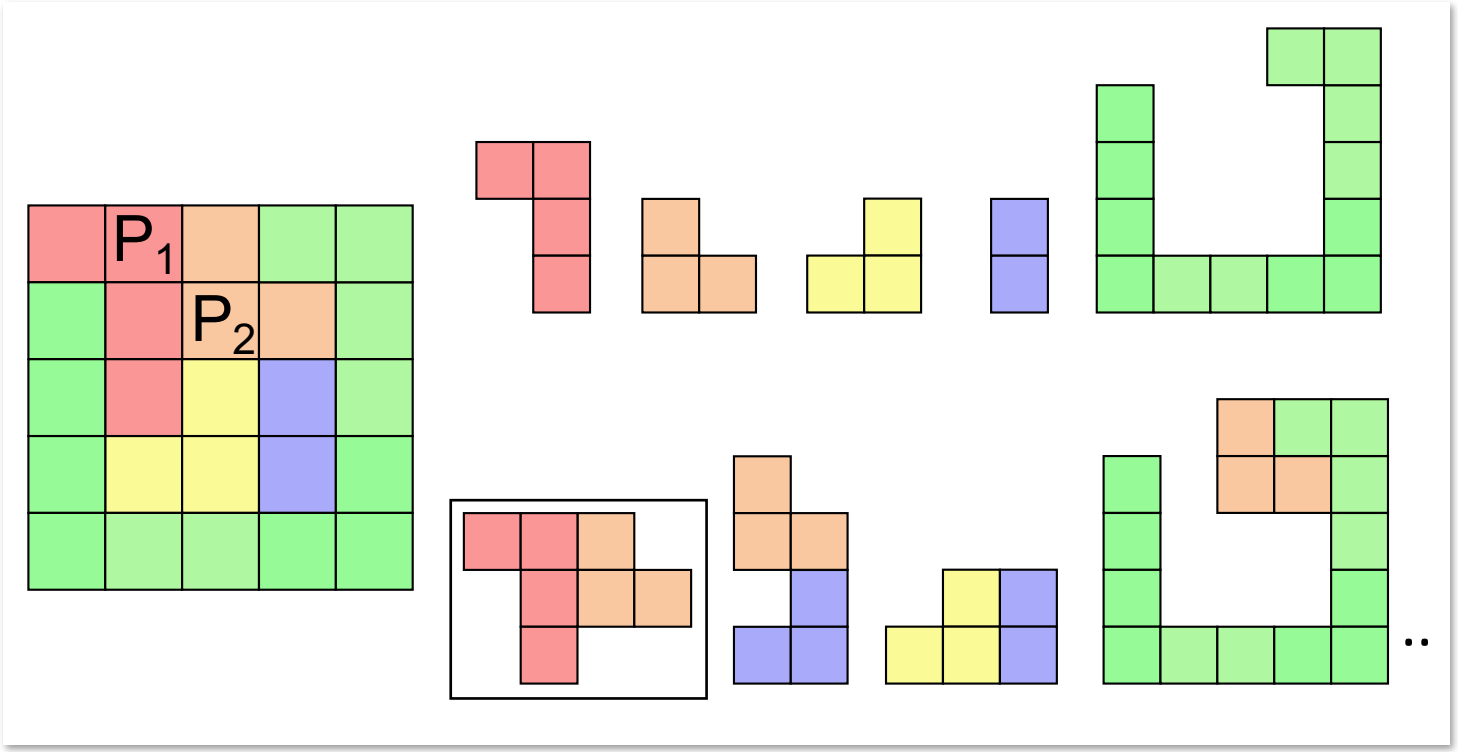
$2^{(K-1)}$ subsets of parts

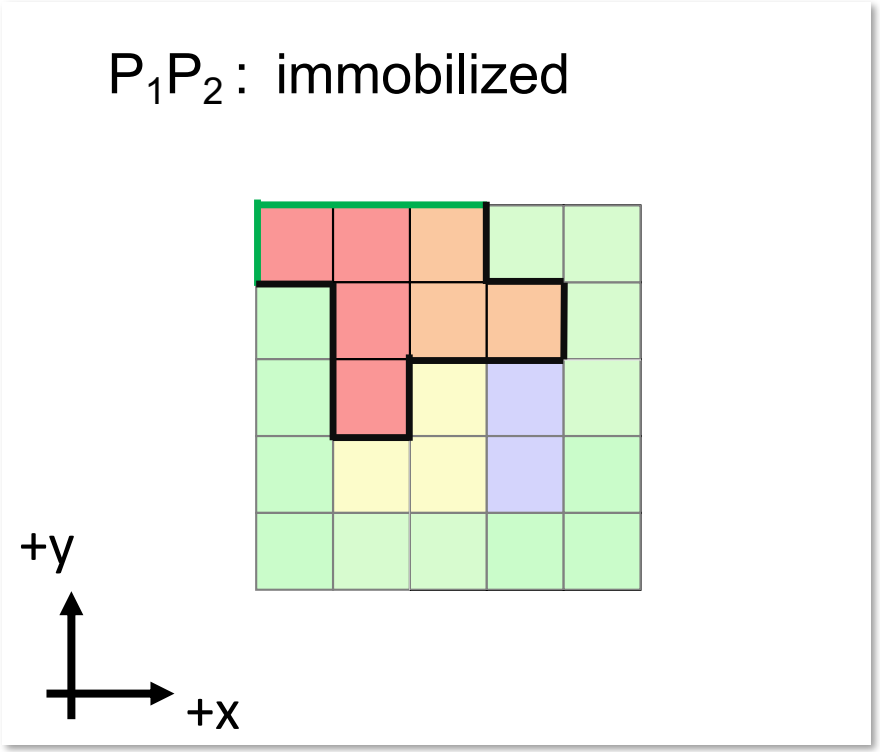


Test mobility of 4 axial directions

Test Interlocking: Parts Immobilization

An assembly is interlocking if only one movable part (key), while all other parts, as well as any subset of parts, are immobilized



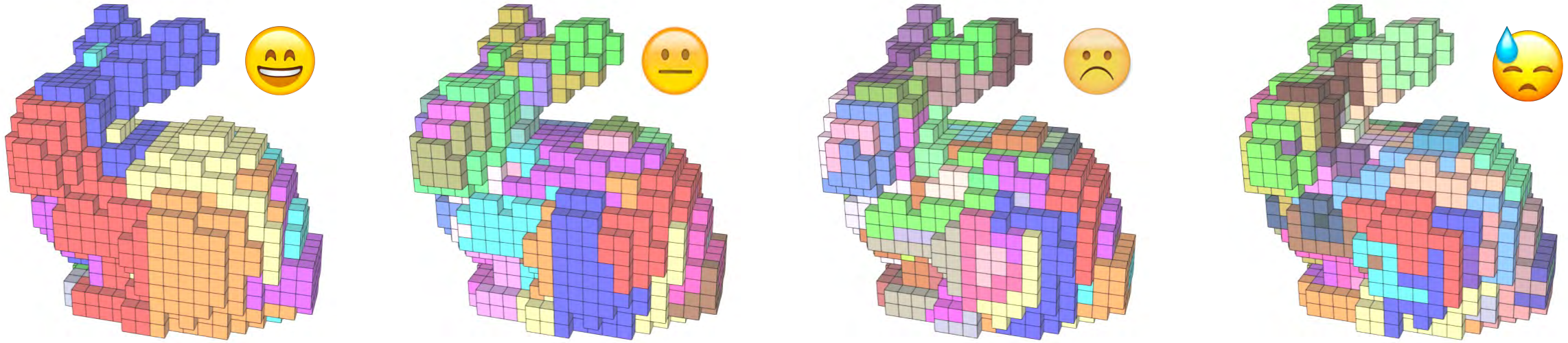


Total cost = $2^{(K-1)}$ subsets of parts × Test mobility of 4 axial directions

Test Interlocking: Parts Immobilization

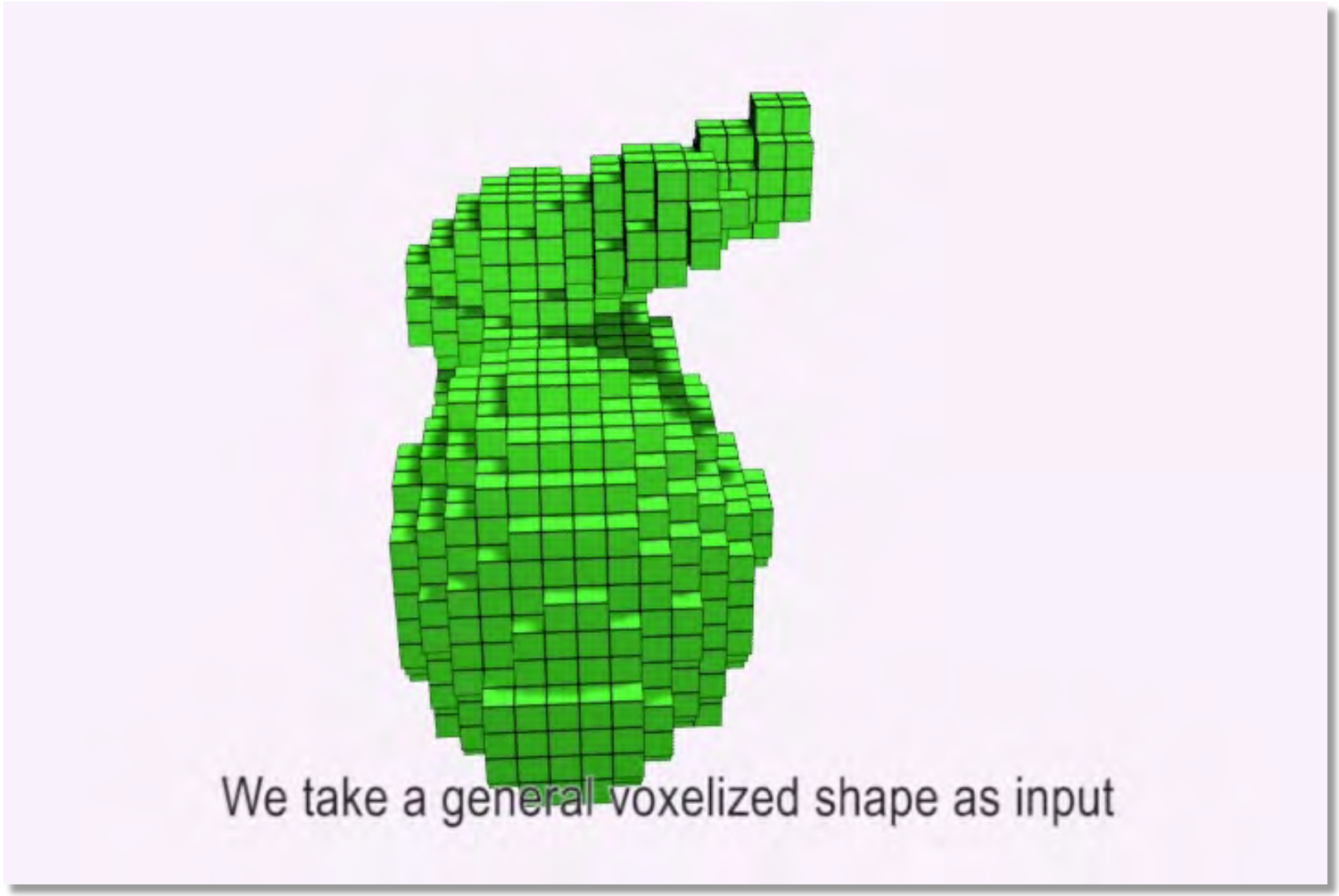
The complexity of testing interlocking is exponential in K

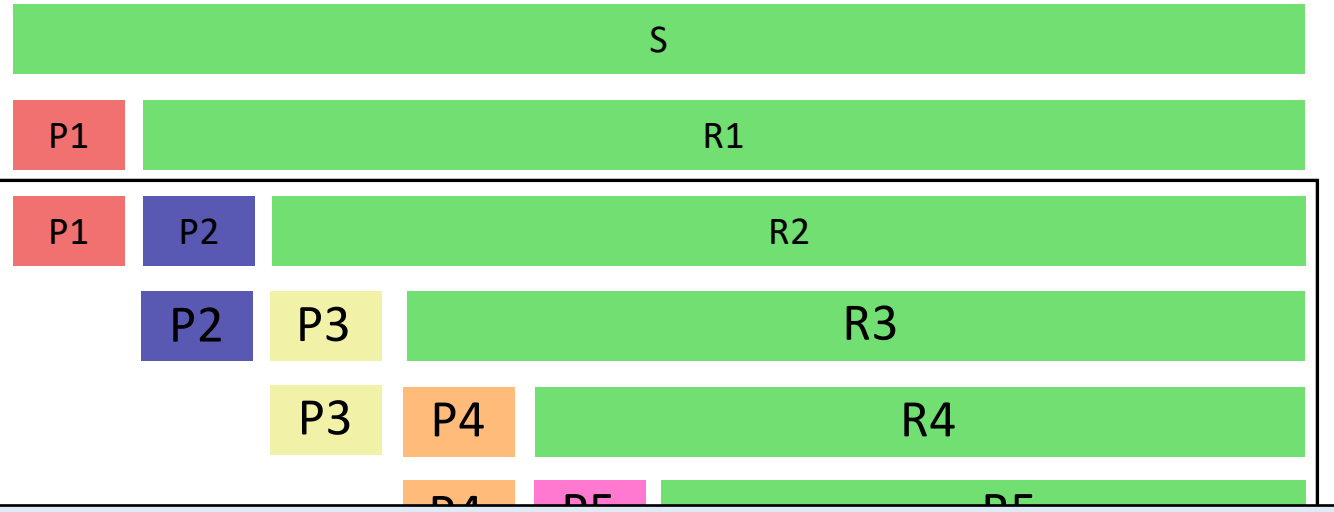
K = 10 K = 20 K = 30 K = 40

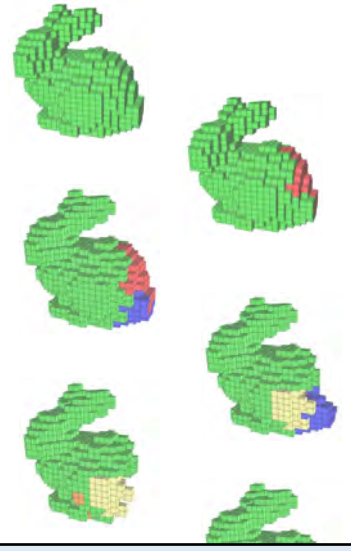


Challenge : Test interlocking for puzzle with K > 20 parts is too computational expensive;
Yet, we want to design interlocking puzzles with K as large as possible (e.g., K = 100)

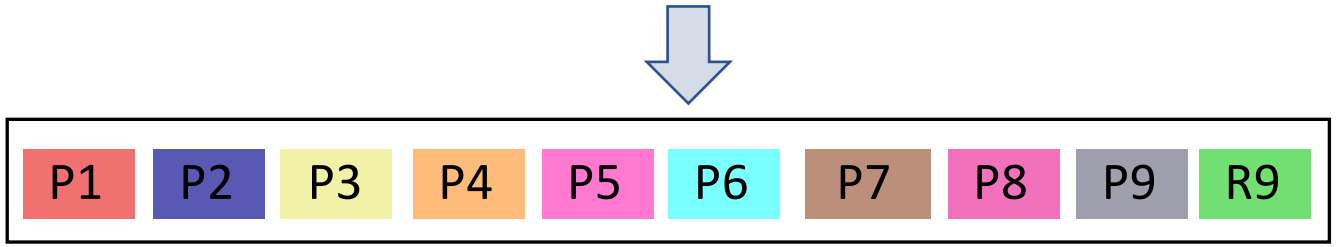
Key Idea #1: Formal Model







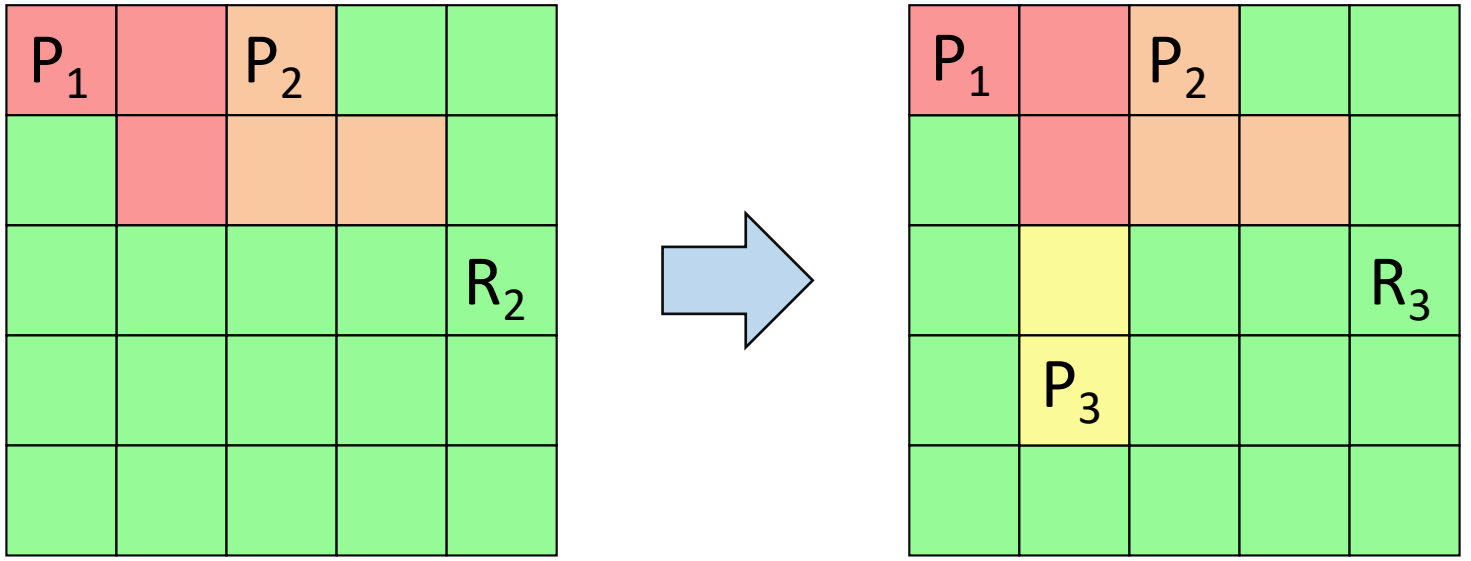
Skip **the exponential time complexity** of testing global interlocking!!!



Key Idea #2: Partition Requirements

Given $[P_1, \ldots, P_{i-1}, R_{i-1}]$, partition P_{i-1} $(i>1)$ into P_i and R_i such that

1. $[P_{i-1}, P_i, R_i]$ is interlocking

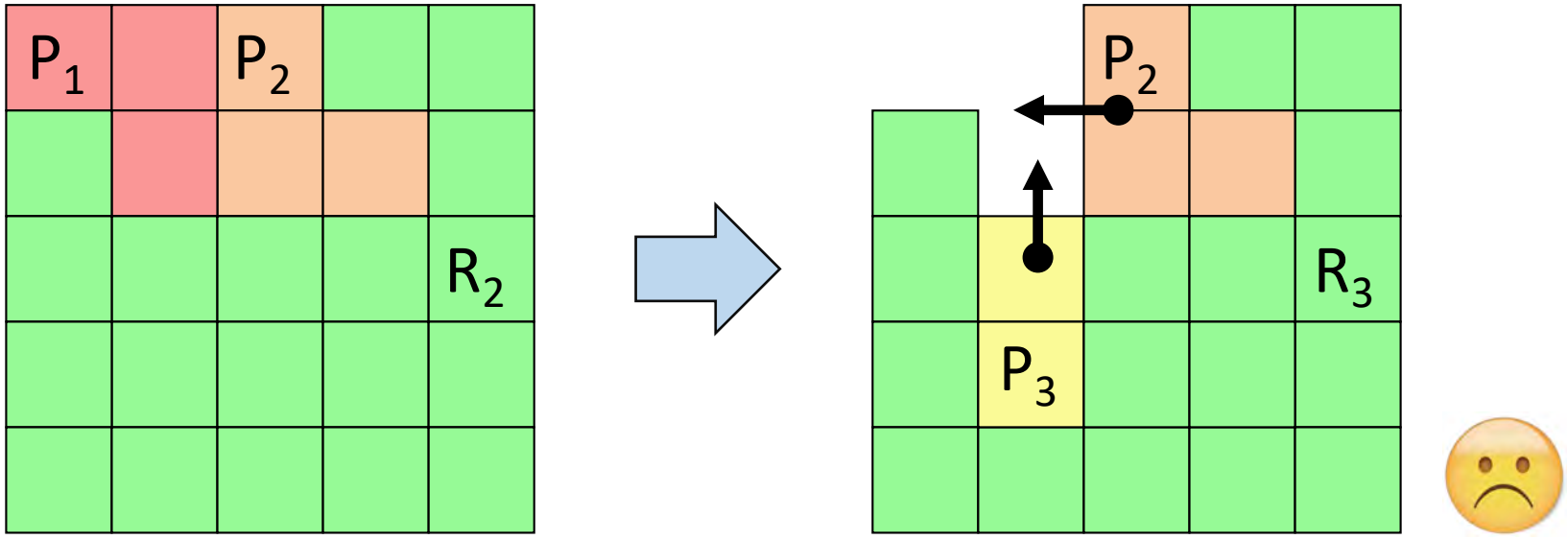


$[P_1, P_2, R_2]$ $[P_1, P_2, P_3, R_3]$

Key Idea #2: Partition Requirements

Given $[P_1, \ldots, P_{i-1}, R_{i-1}]$, partition P_{i-1} $(i>1)$ into P_i and R_i such that

1. $[P_{i-1}, P_i, R_i]$ is interlocking

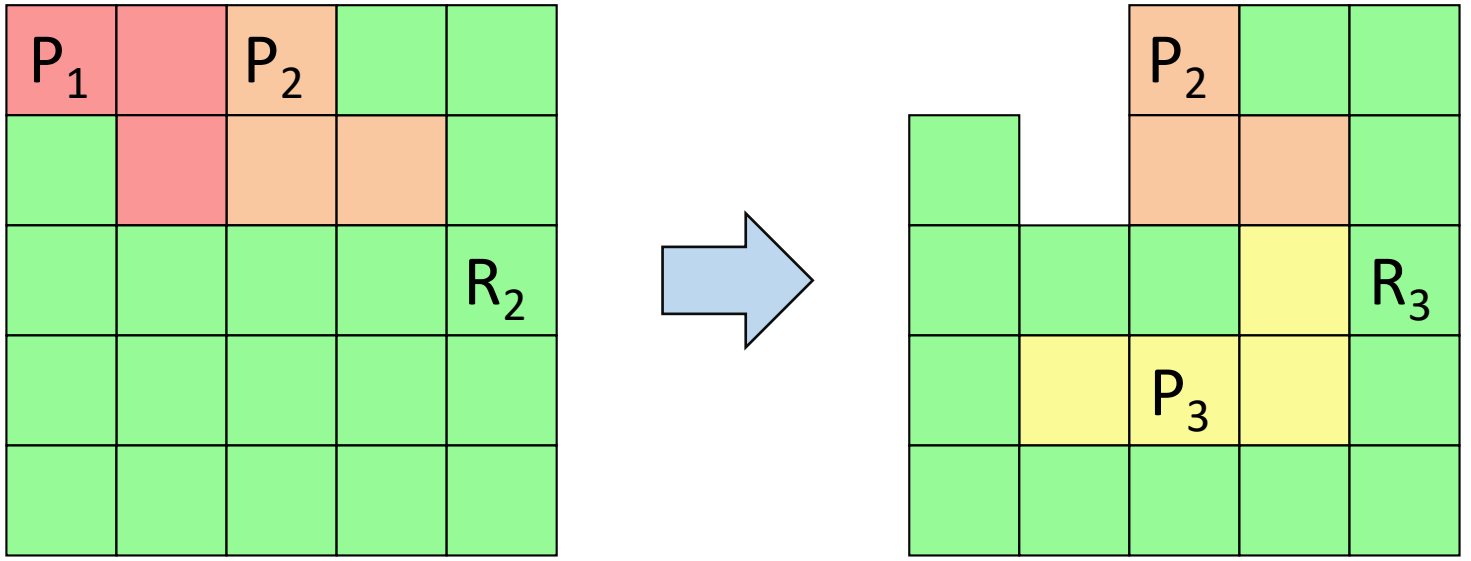


$[P_1, P_2, R_2]$ $[P_1, P_2, P_3, R_3]$

Key Idea #2: Partition Requirements

Given $[P_1, \ldots, P_{i-1}, R_{i-1}]$, partition P_{i-1} $(i>1)$ into P_i and R_i such that

1. $[P_{i-1}, P_i, R_i]$ is interlocking
2. P_i is disassemblable in $[P_i, R_i]$

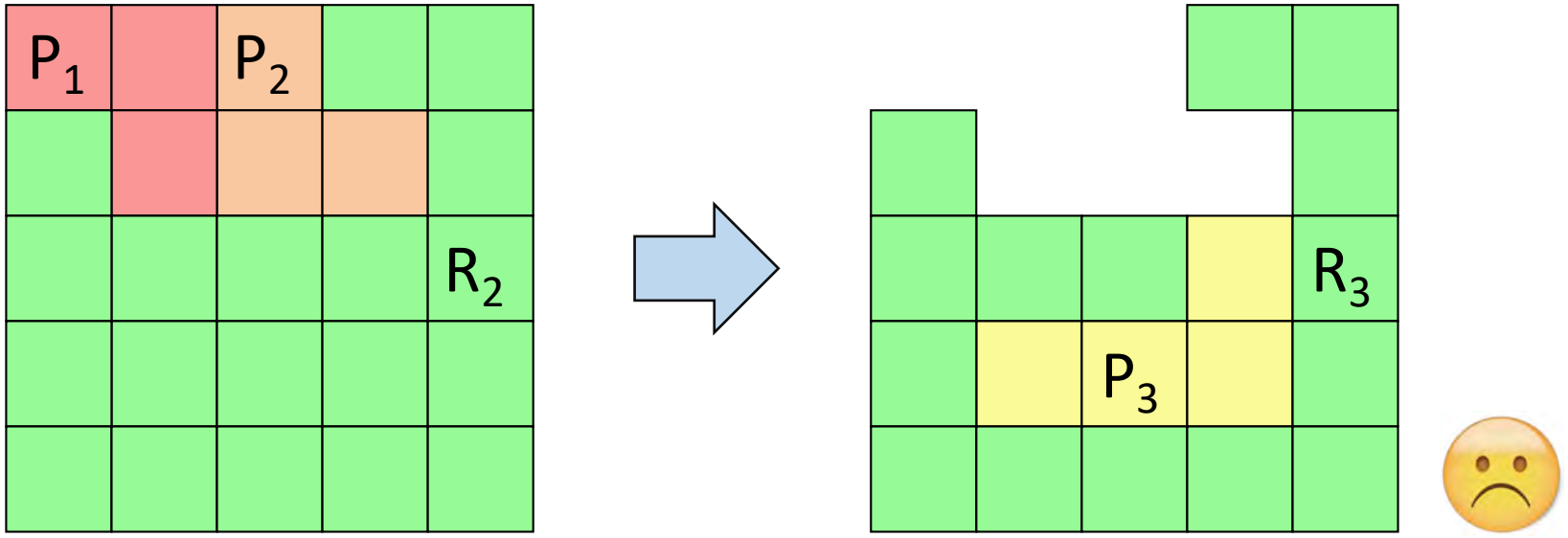


$[P_1, P_2, R_2]$ $[P_1, P_2, P_3, R_3]$

Key Idea #2: Partition Requirements

Given $[P_1, \ldots, P_{i-1}, R_{i-1}]$, partition P_{i-1} $(i>1)$ into P_i and R_i such that

1. $[P_{i-1}, P_i, R_i]$ is interlocking
2. P_i is disassemblable in $[P_i, R_i]$

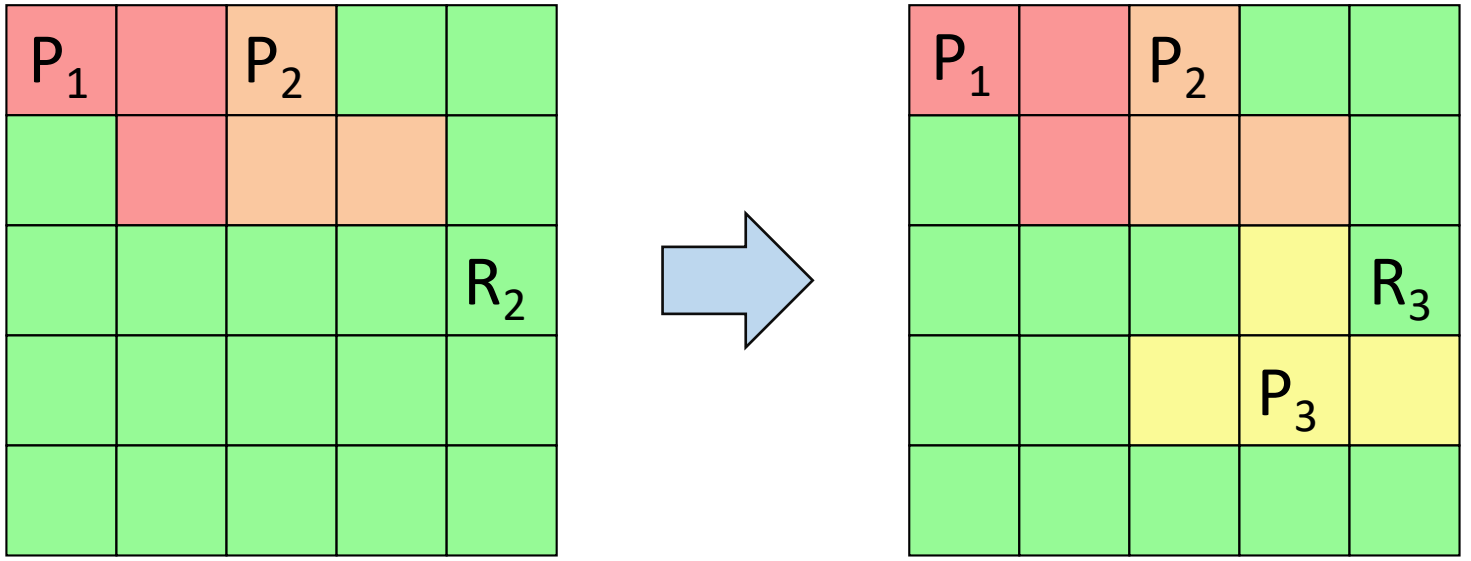


$[P_1, P_2, R_2]$ $[P_1, P_2, P_3, R_3]$

Key Idea #2: Partition Requirements

Given $[P_1, \ldots, P_{i-1}, R_{i-1}]$, partition P_{i-1} $(i>1)$ into P_i and R_i such that

1. $[P_{i-1}, P_i, R_i]$ is interlocking
2. P_i is disassemblable in $[P_i, R_i]$
3. P_i is connected; R_i is connected

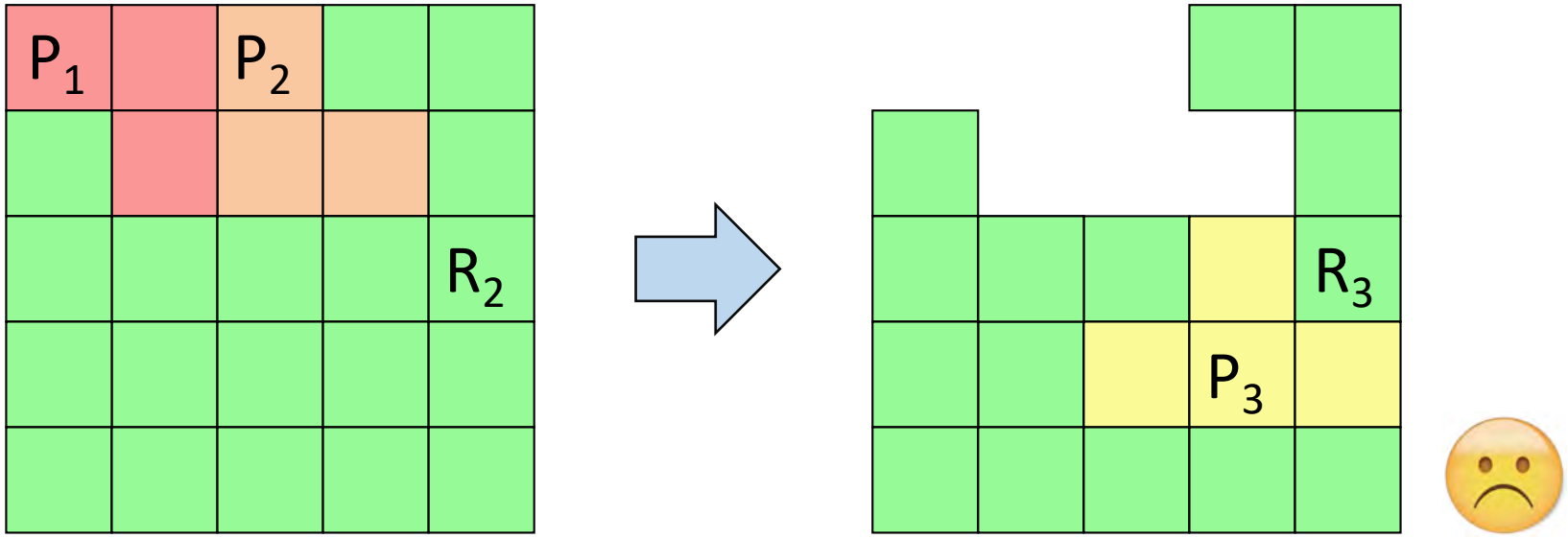


$[P_1, P_2, R_2]$ $[P_1, P_2, P_3, R_3]$

Key Idea #2: Partition Requirements

Given $[P_1, \ldots, P_{i-1}, R_{i-1}]$, partition P_{i-1} $(i>1)$ into P_i and R_i such that

1. $[P_{i-1}, P_i, R_i]$ is interlocking
2. P_i is disassemblable in $[P_i, R_i]$
3. P_i is connected; R_i is connected

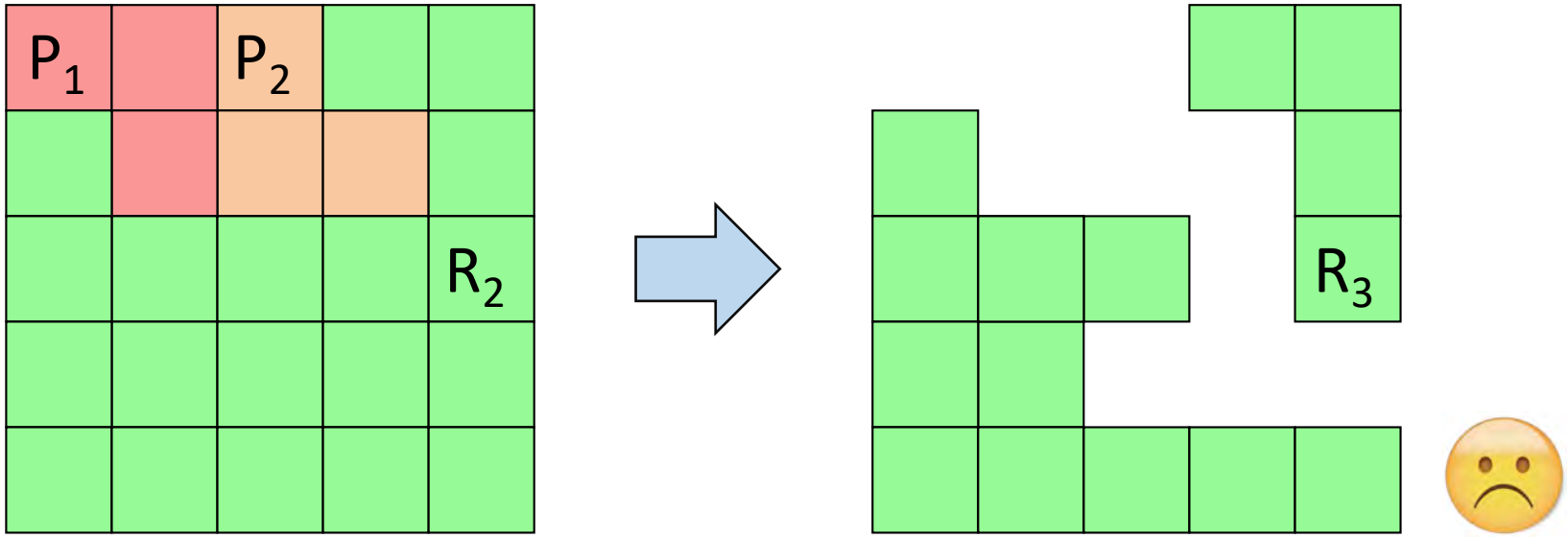


$[P_1, P_2, R_2]$ $[P_1, P_2, P_3, R_3]$

Key Idea #2: Partition Requirements

Given $[P_1, \ldots, P_{i-1}, R_{i-1}]$, partition P_{i-1} $(i>1)$ into P_i and R_i such that

1. $[P_{i-1}, P_i, R_i]$ is interlocking
2. P_i is disassemblable in $[P_i, R_i]$
3. P_i is connected; R_i is connected

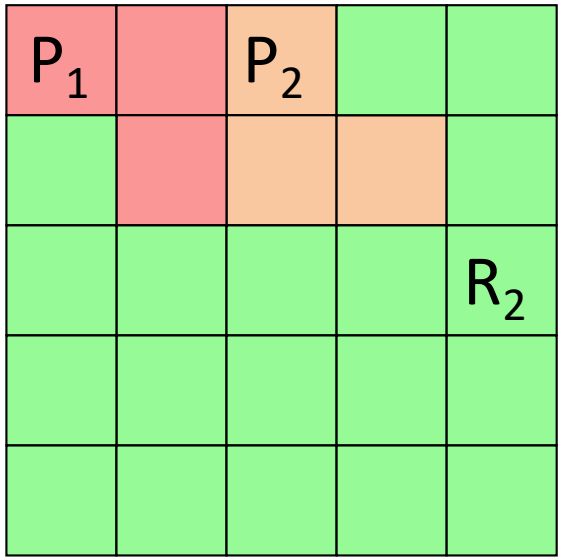


$[P_1, P_2, R_2]$ $[P_1, P_2, P_3, R_3]$

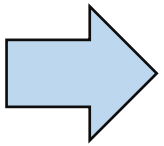
Key Idea #2: Partition Requirements

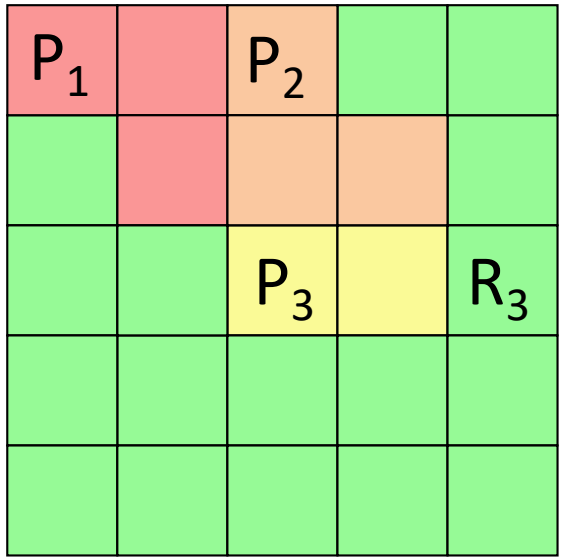
Given $[P_1, \ldots, P_{i-1}, R_{i-1}]$, partition P_{i-1} $(i>1)$ into P_i and R_i such that

1. $[P_{i-1}, P_i, R_i]$ is interlocking
2. P_i is disassemblable in $[P_i, R_i]$
3. P_i is connected; R_i is connected



$[P_1, P_2, R_2]$

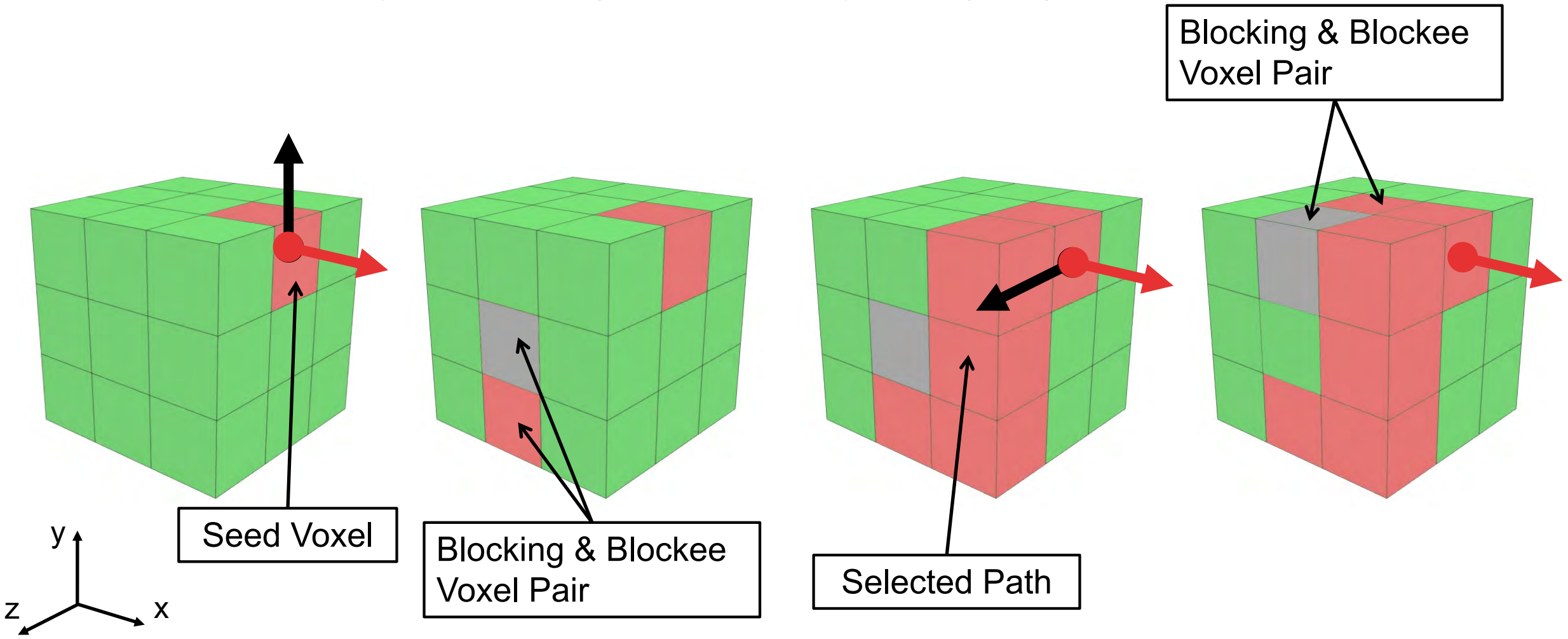




$[P_1, P_2, P_3, R_3]$

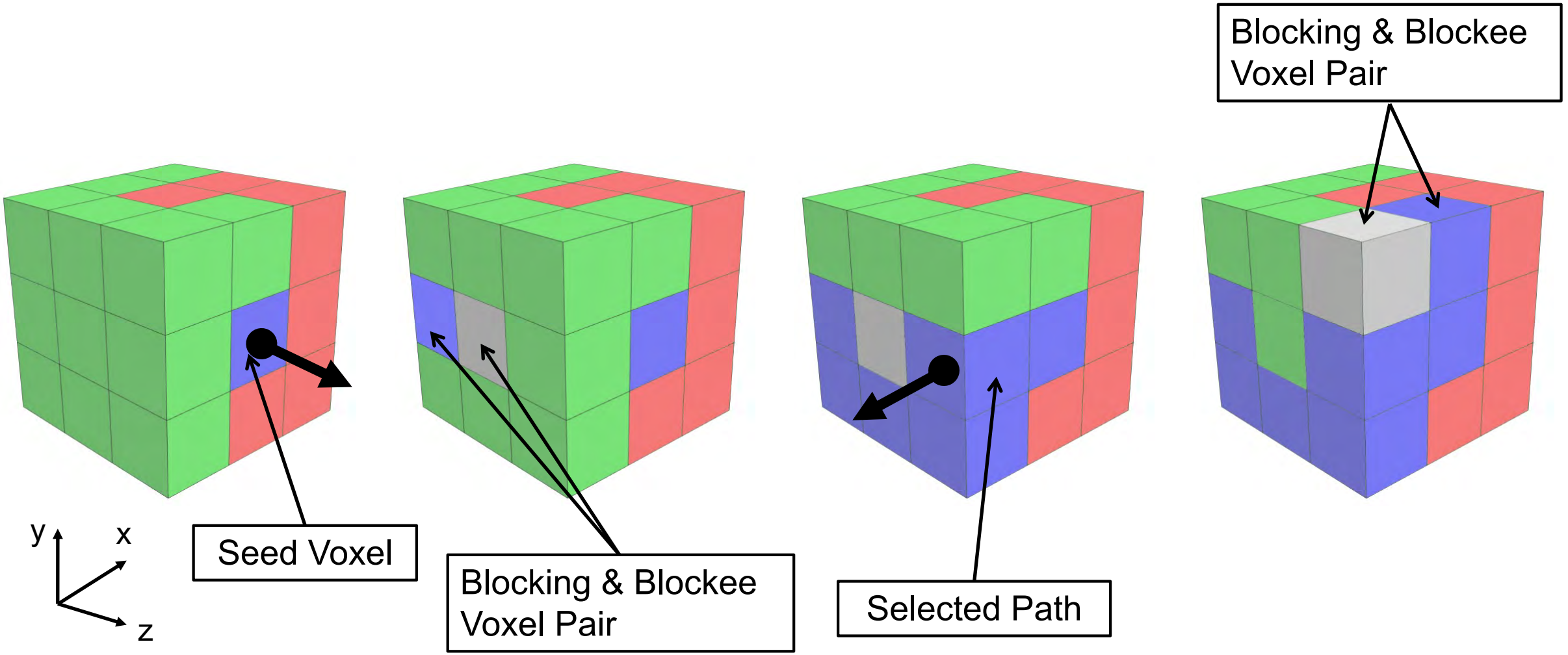
Key Idea #3: Constructive Approach

Construct the key piece P_1 (movable only along +x)

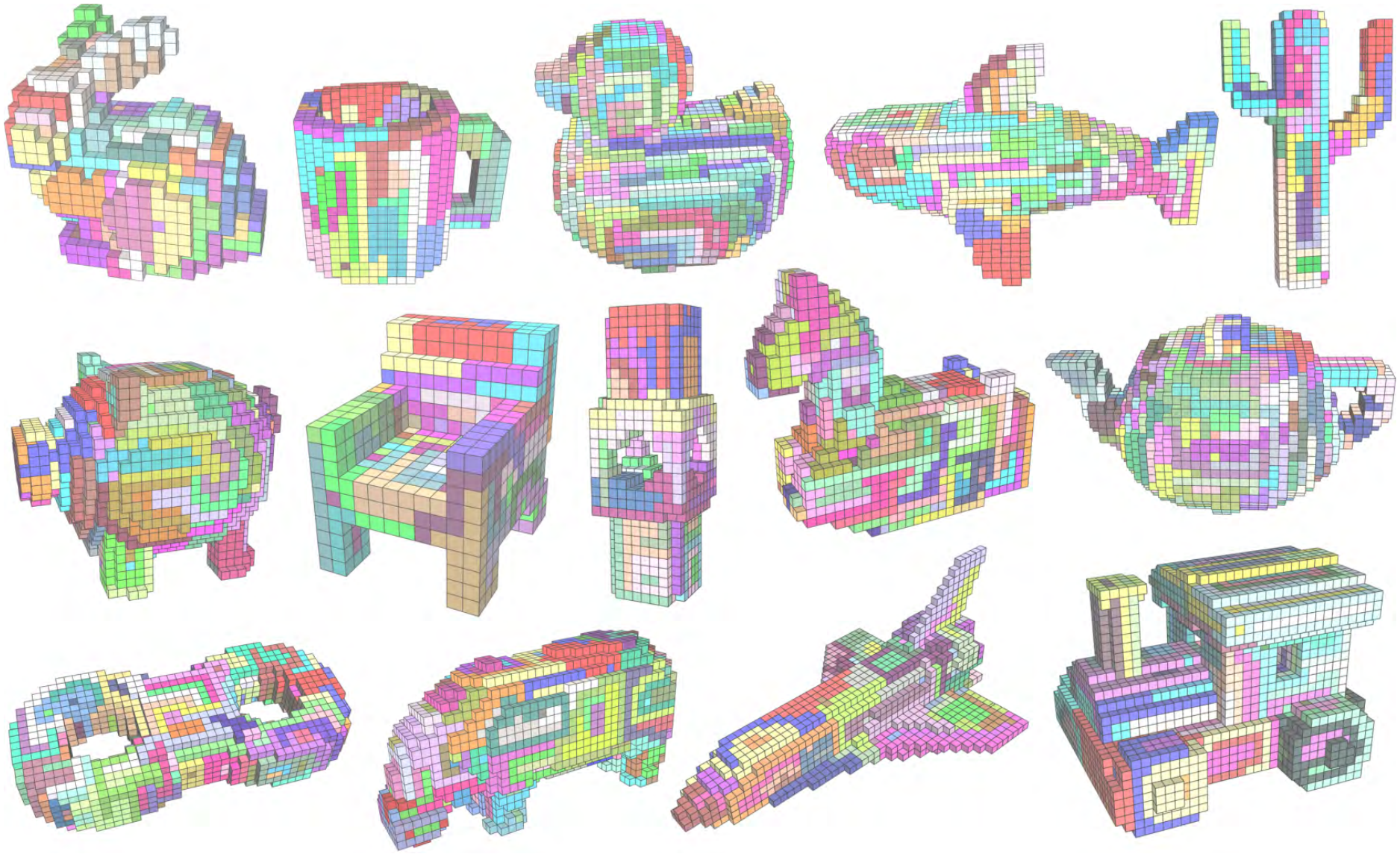


Key Idea #3: Constructive Approach

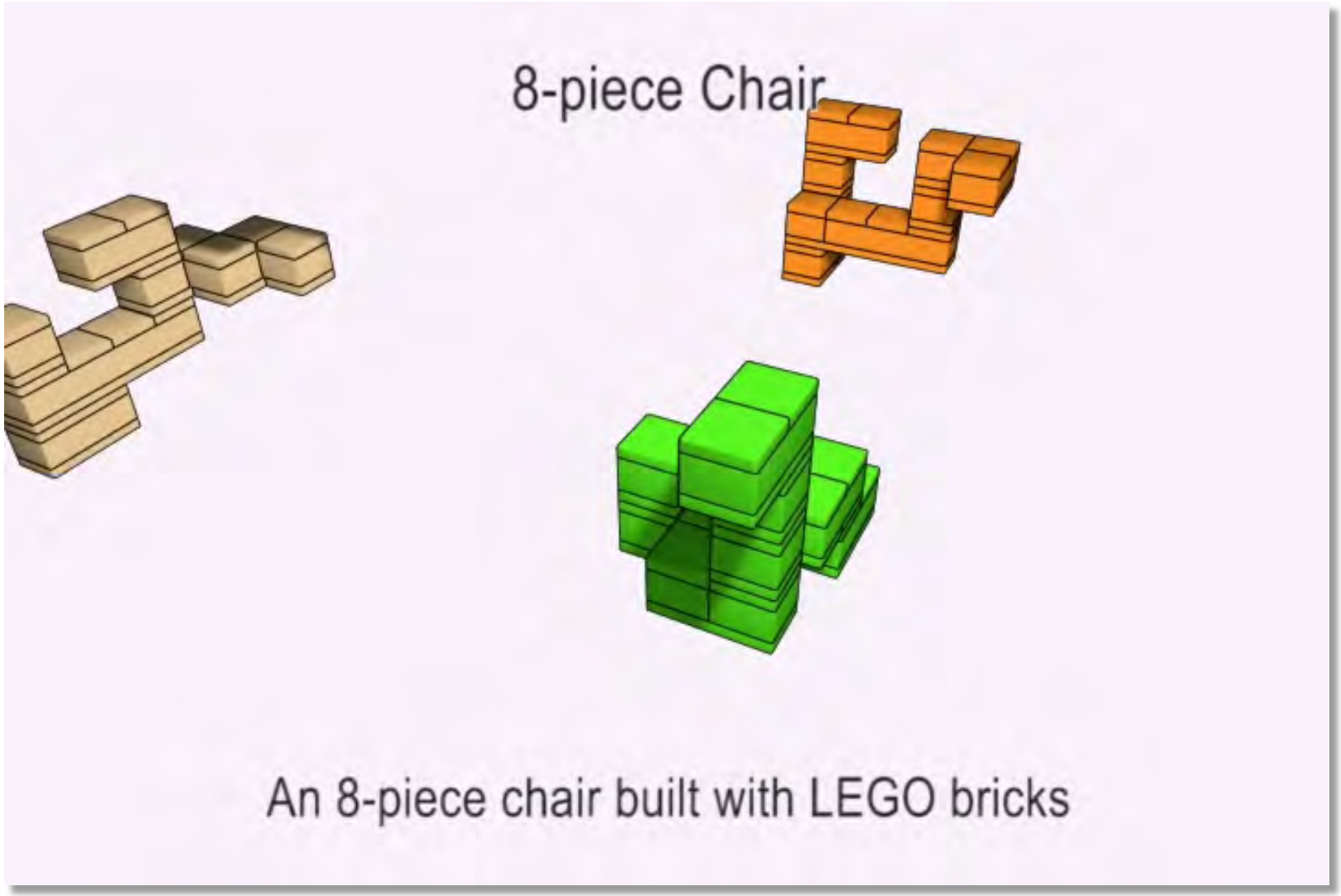
Construct the 2nd piece P_2 (immobilized by the key)



Our Result



Our Result



Our Result

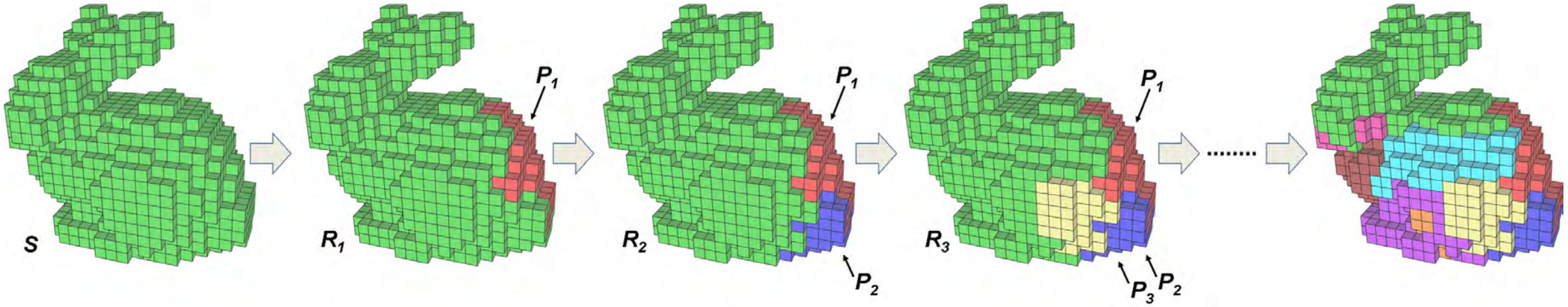
115-piece Shark

A wooden shark puzzle.

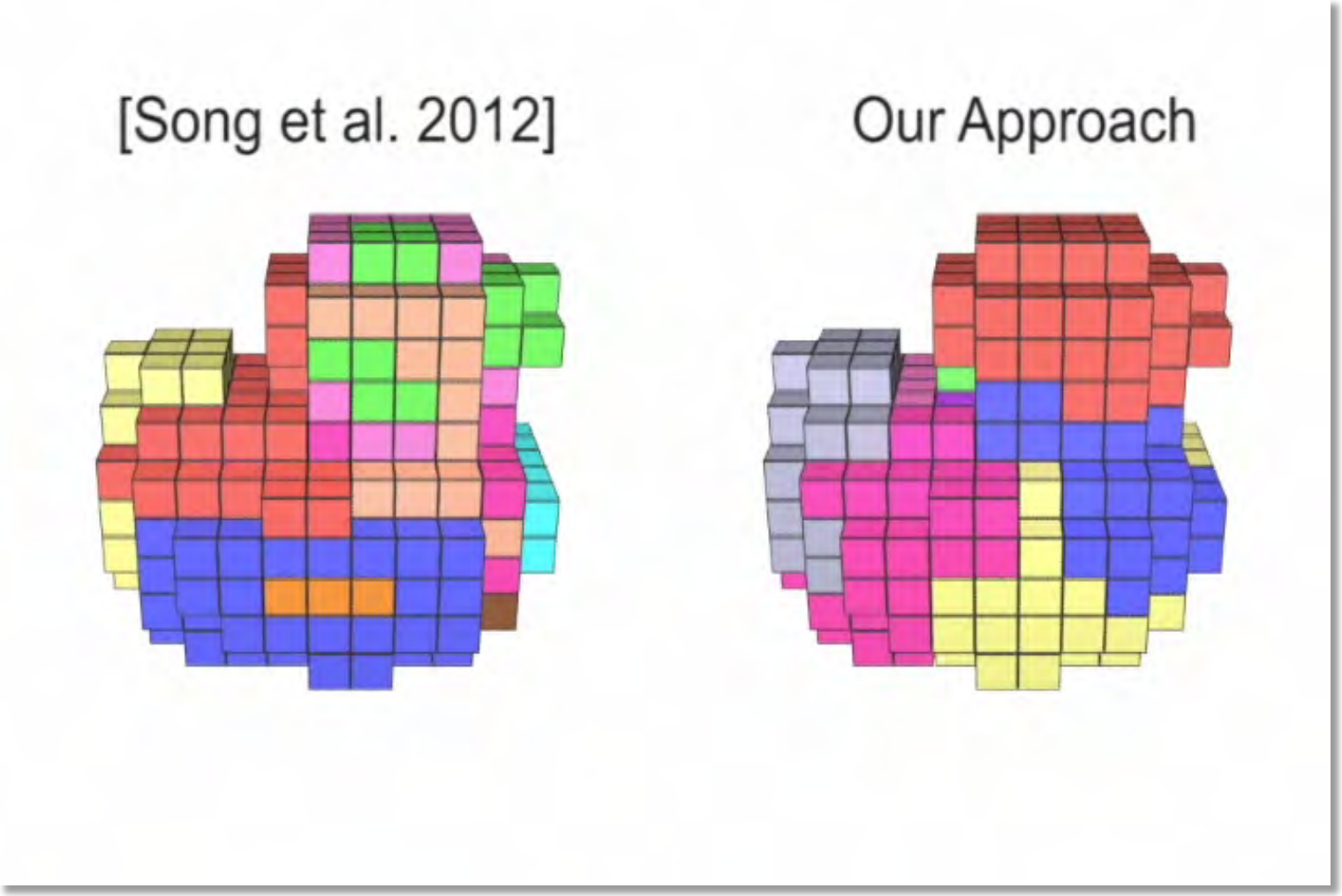
Our Result

Summary of the project

- A **formal model** to directly guarantee recursive interlocking based on building local interlocking groups (LIGs)
- **Requirements** to ensure local interlocking of intermediate assemblies when extracting each puzzle piece
- A **constructive approach** to iteratively generate geometry of each puzzle piece

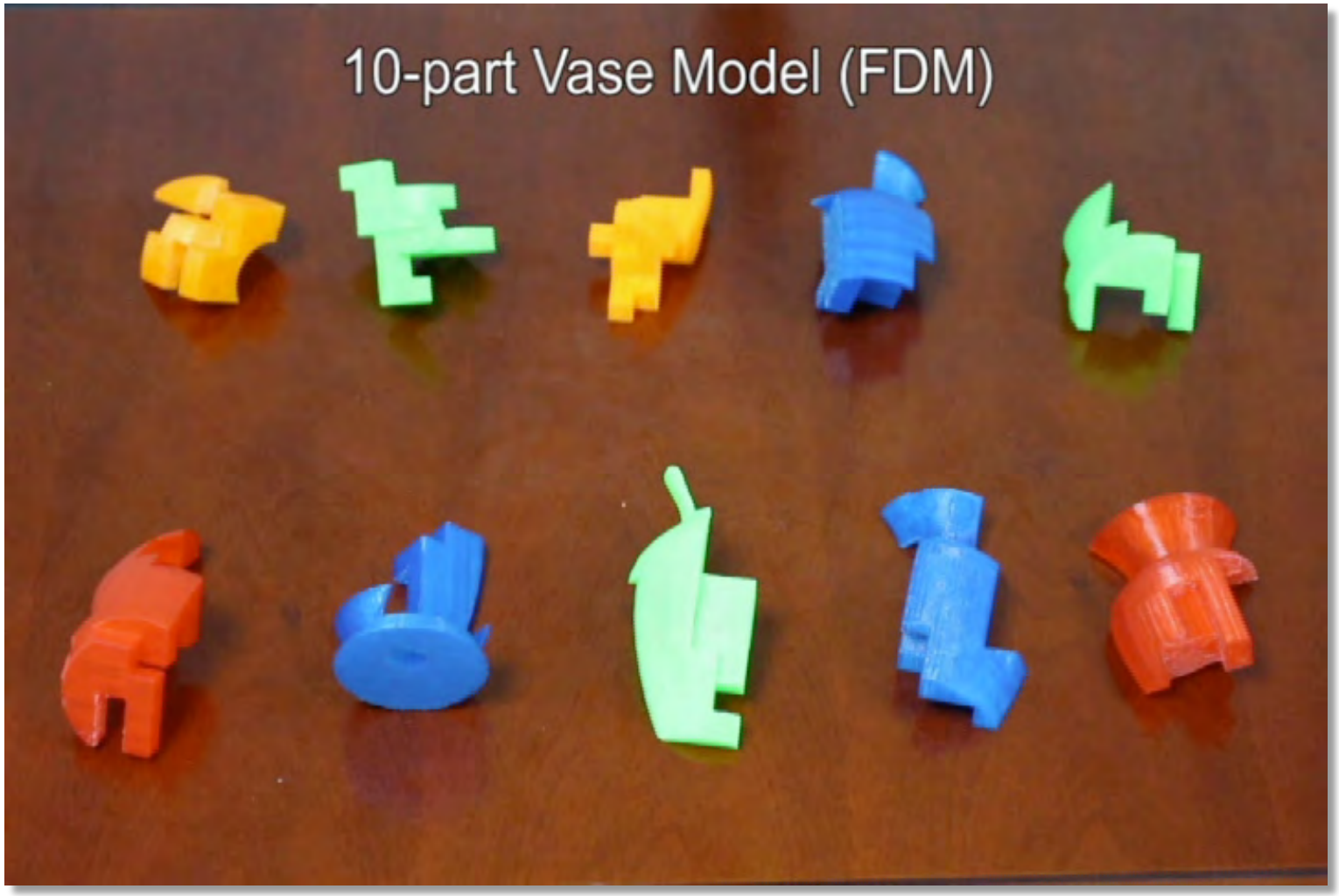


Follow-up Work: Interlocking Objects for 3D Printing



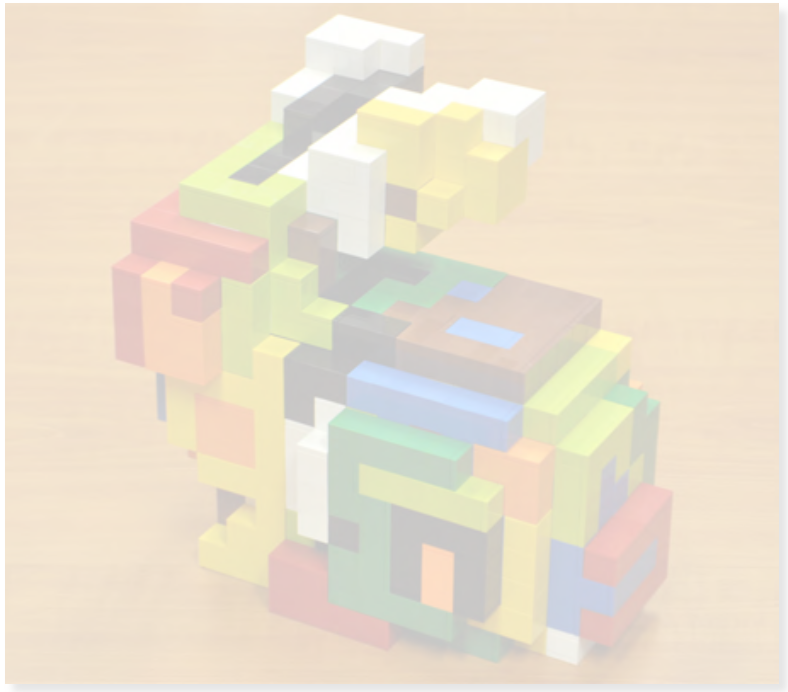
Song et al. *Printing 3D Objects with Interlocking Parts*. CAGD (Proc. of GMP), 2015

Follow-up Work: Interlocking Objects for 3D Printing



Song et al. *Printing 3D Objects with Interlocking Parts*. CAGD (Proc. of GMP), 2015

Overview



Recursive Interlocking Puzzles

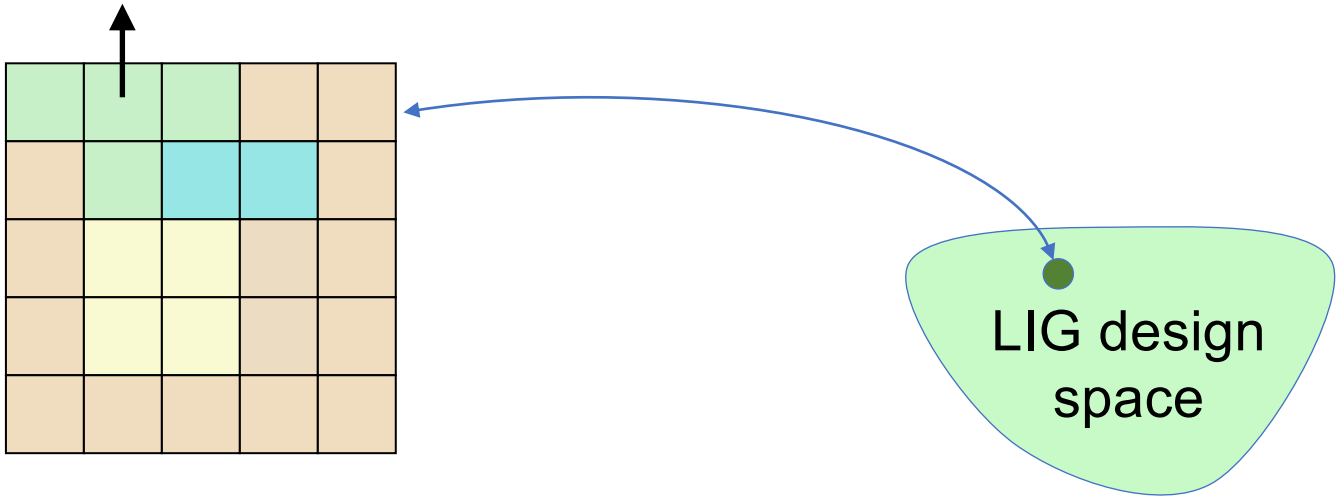
SIGGRAPH Asia 2012

DESIA: A General Framework for Designing Interlocking Assemblies

SIGGRAPH Asia 2018

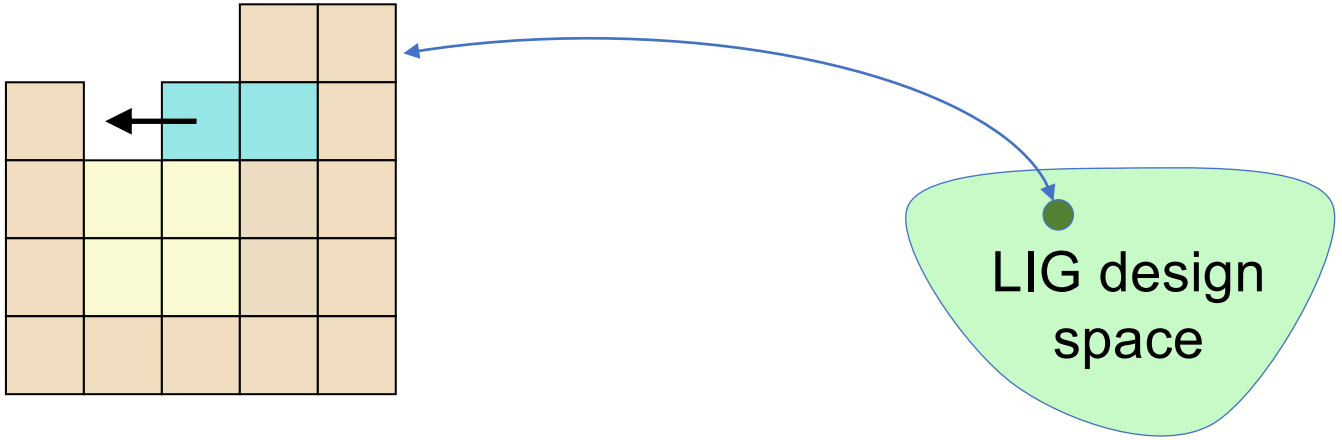
Motivation

The Recursive Interlocking approach can explore only a limited design space



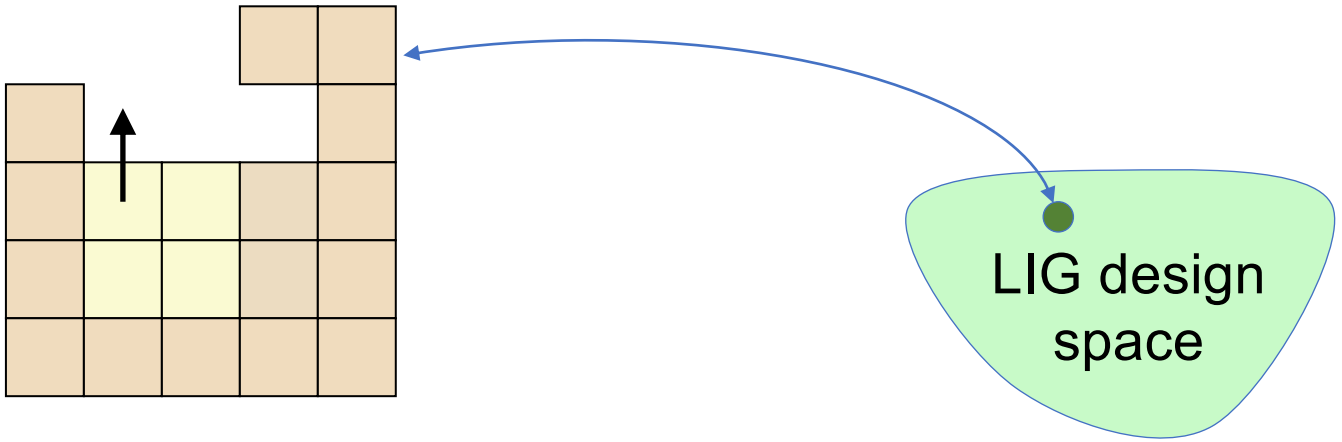
Motivation

The Recursive Interlocking approach can explore only a limited design space



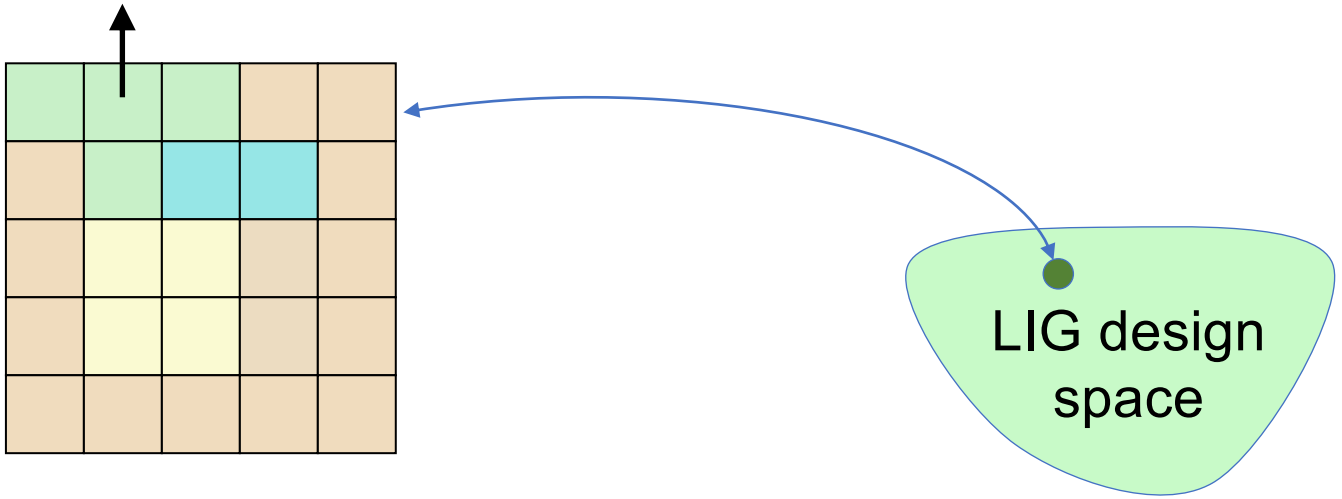
Motivation

The Recursive Interlocking approach can explore only a limited design space



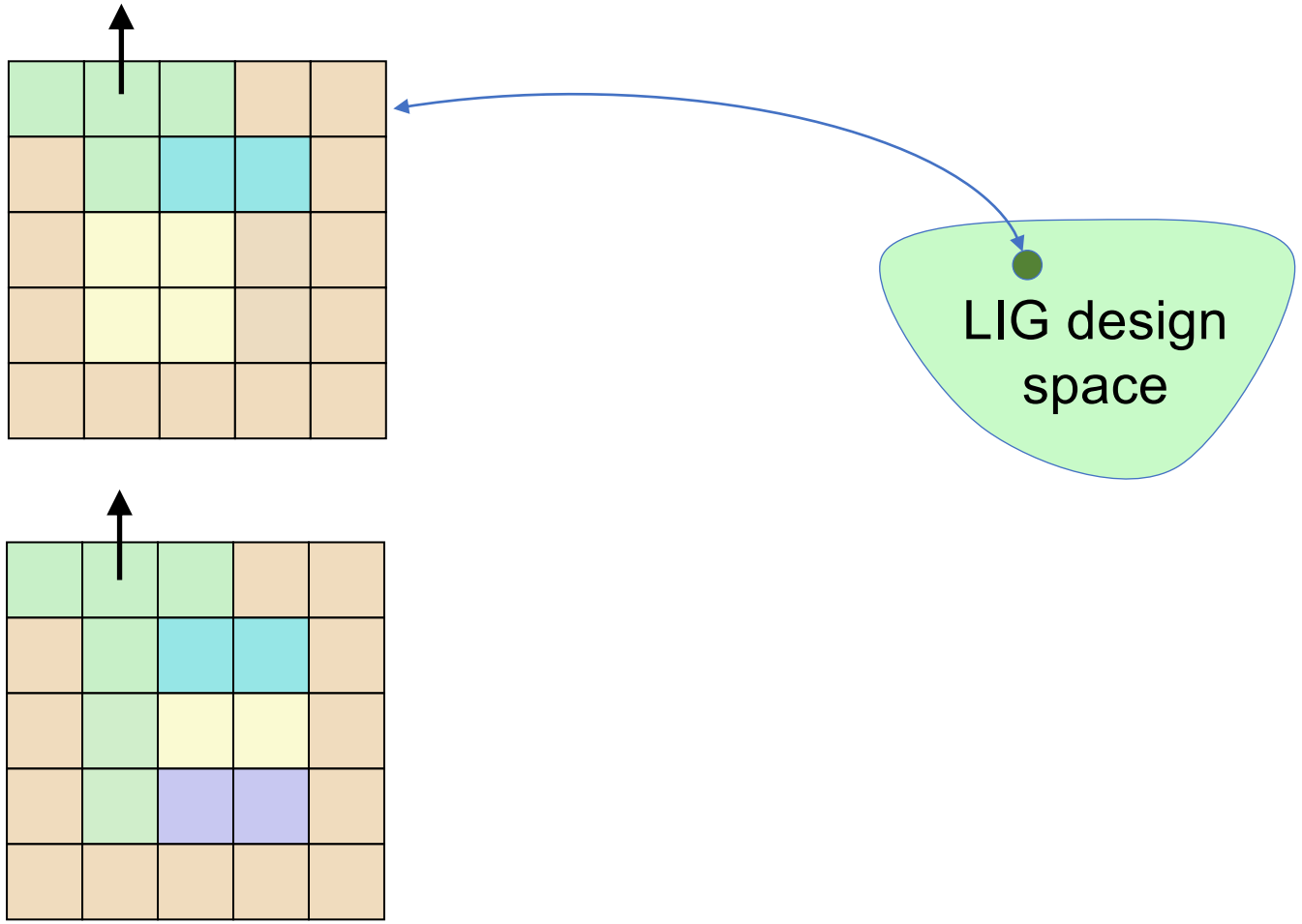
Motivation

The Recursive Interlocking approach can explore only a limited design space



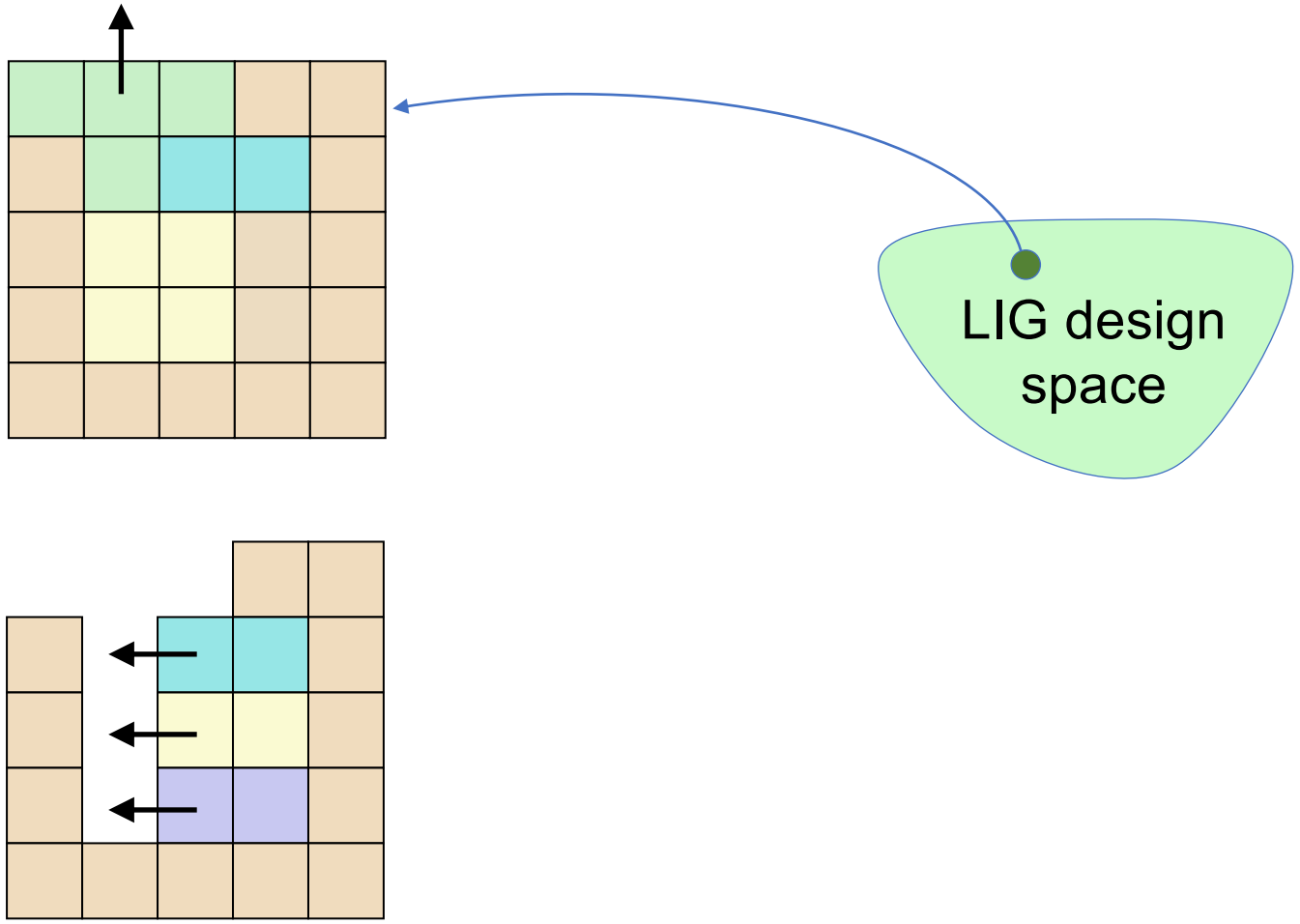
Motivation

The Recursive Interlocking approach can explore only a limited design space



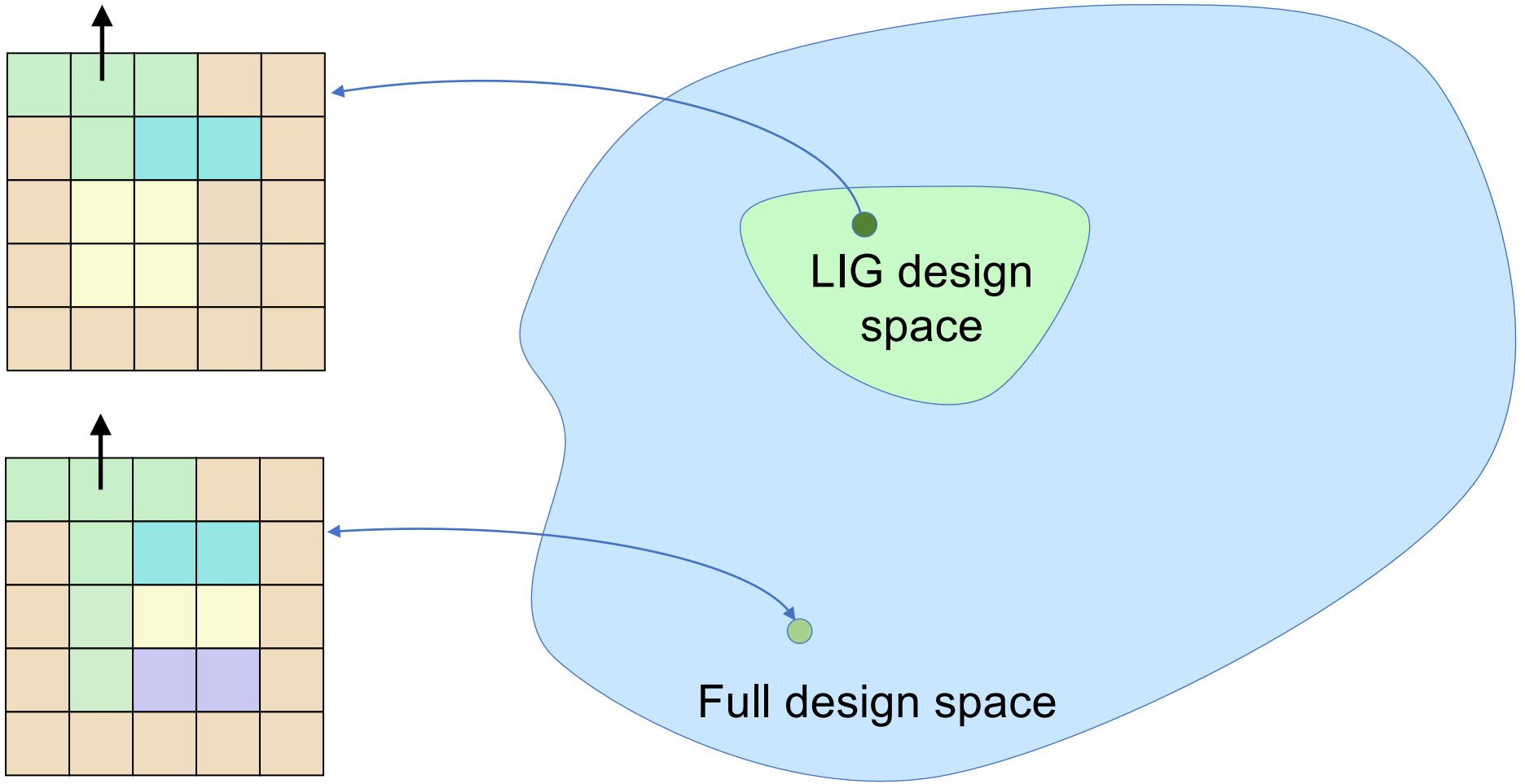
Motivation

The Recursive Interlocking approach can explore only a limited design space



Motivation

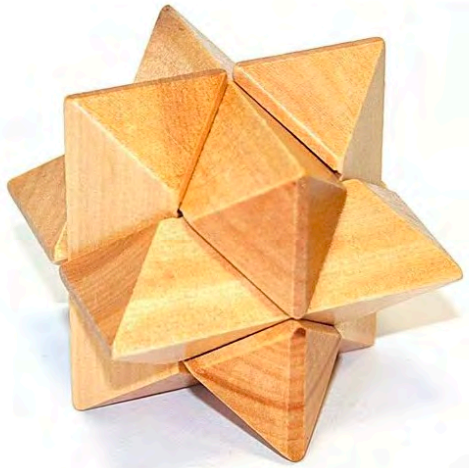
The Recursive Interlocking approach can explore only a limited design space



Our Goal: Design Interlocking Assembly

Can we have a general framework to design interlocking assemblies that can explore the full search space of all possible interlocking configurations?

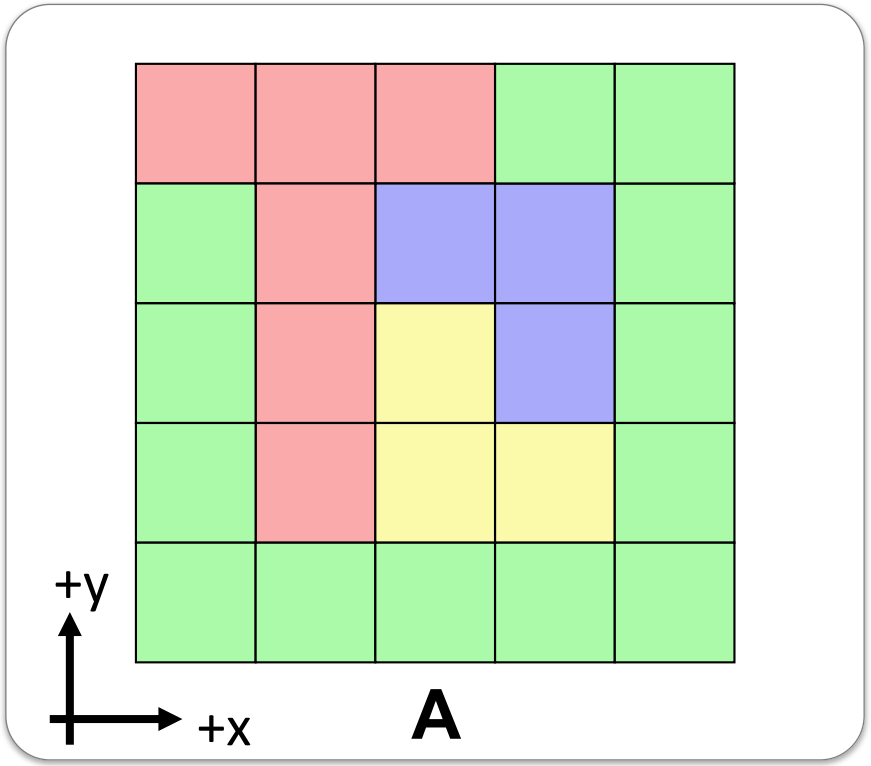
1. Provide more design flexibility
2. Useful for designing new interlocking assemblies

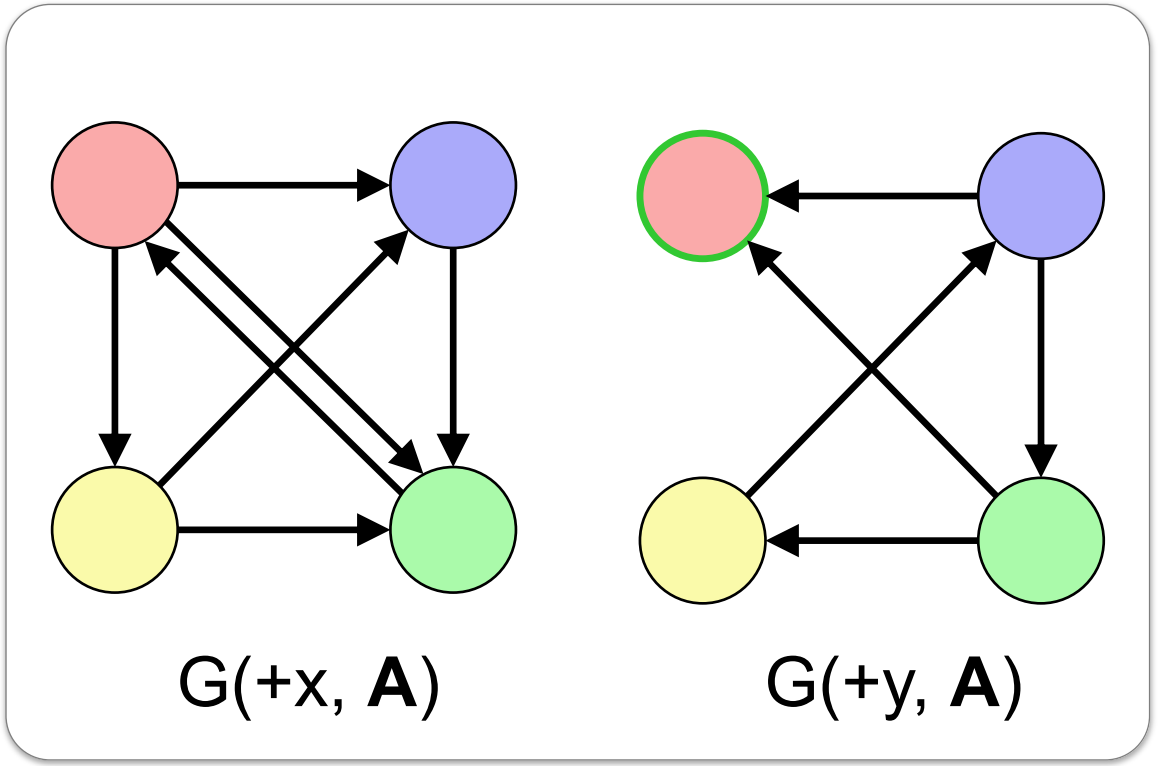
...

Our Key Idea: Graph-based Representation

Test and design
interlocking assemblies



Directional blocking graphs
Invented by Wilson [1992]



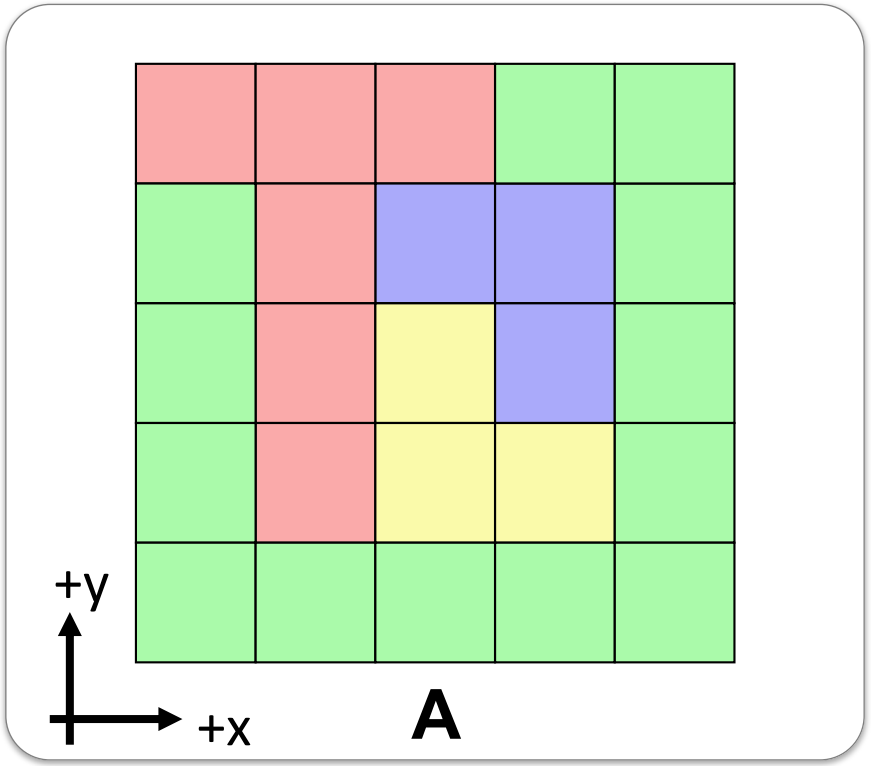
Contribution #1: Test Interlocking

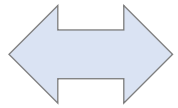
Polynomial time complexity!!!

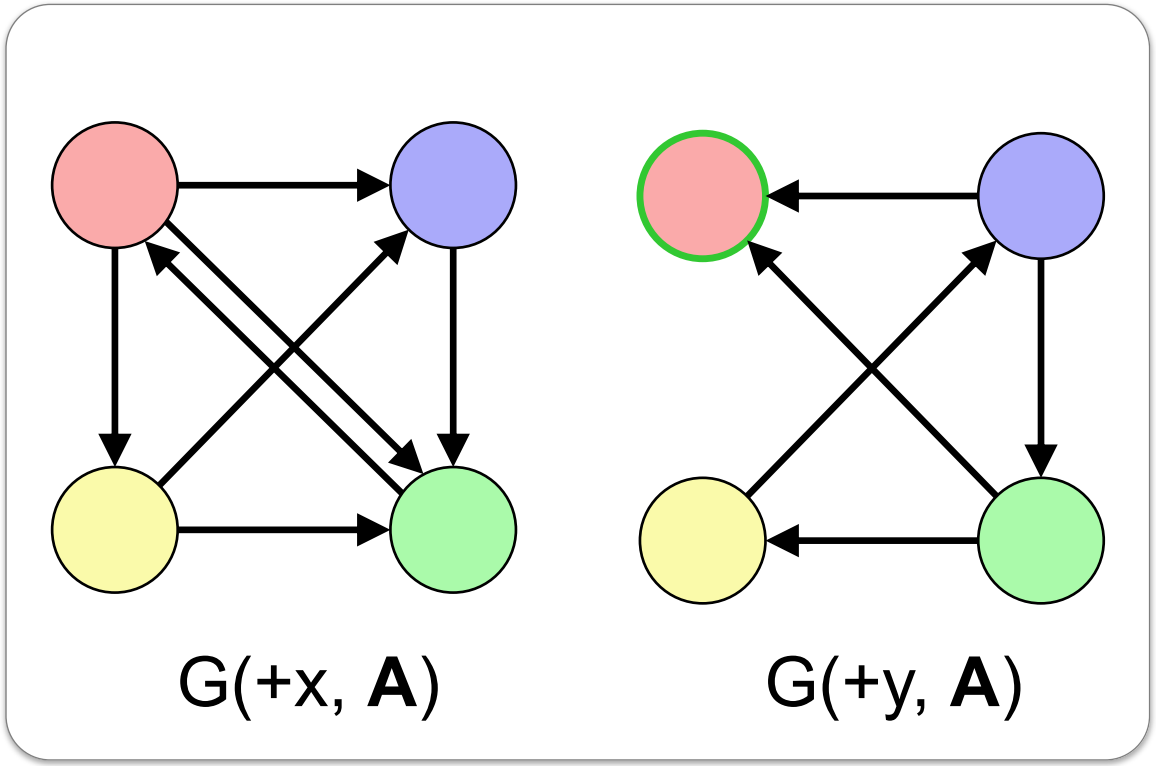
The 3D assembly is interlocking

=

All graphs are strongly connected
(except the key part)

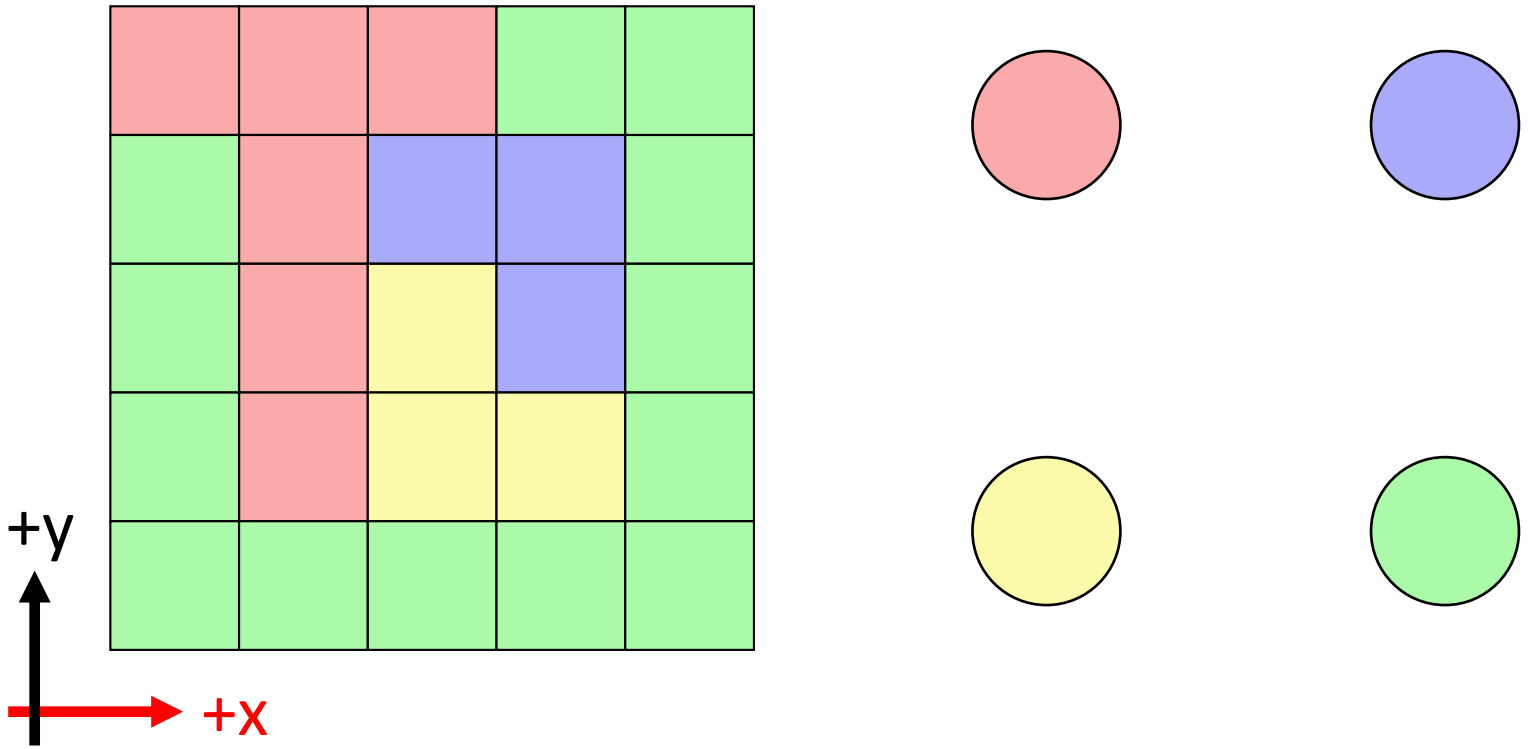






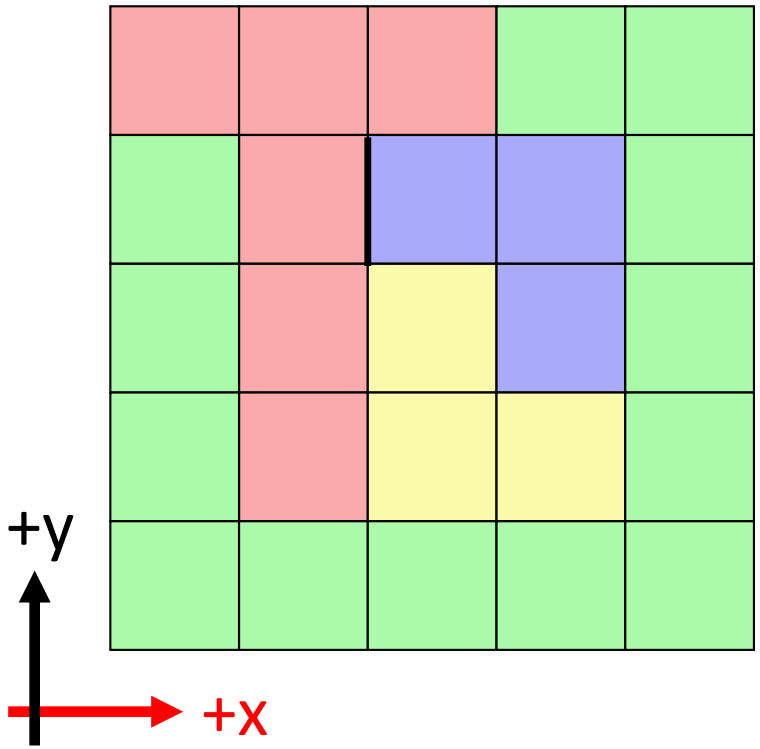
Test Interlocking: Directional Blocking Graph

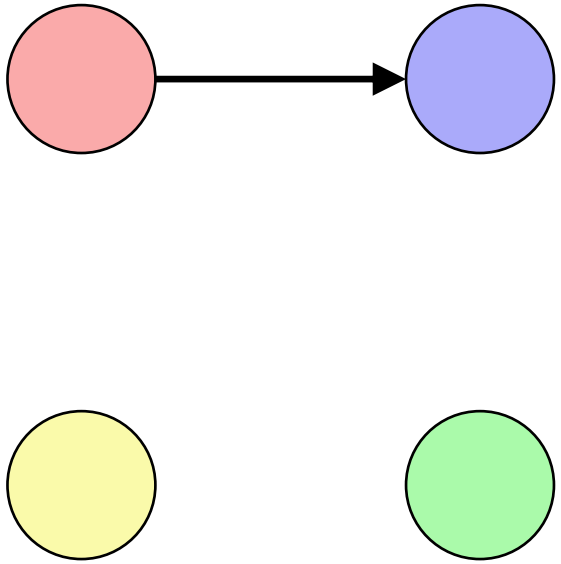
Given an assembly **A** and a certain axial direction *d*



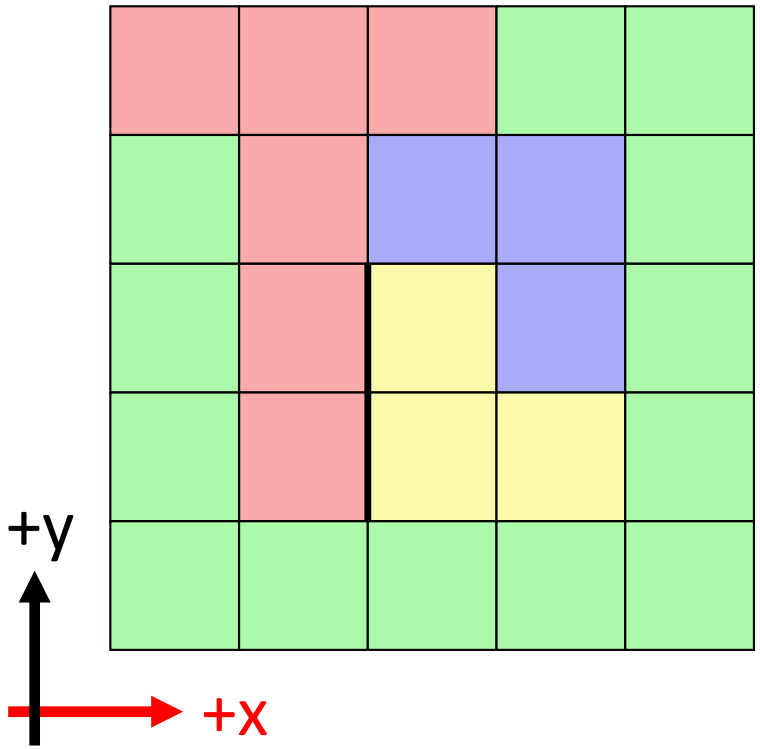
Test Interlocking: Directional Blocking Graph

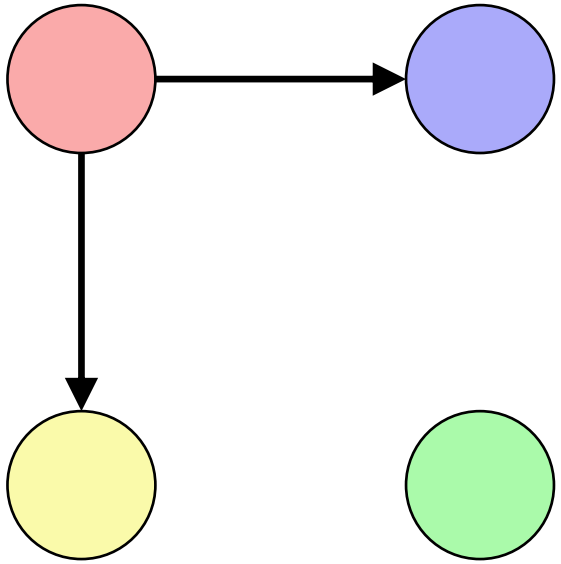
Create a directed edge from P_i to P_j iff P_j blocks P_i from translating along *d*



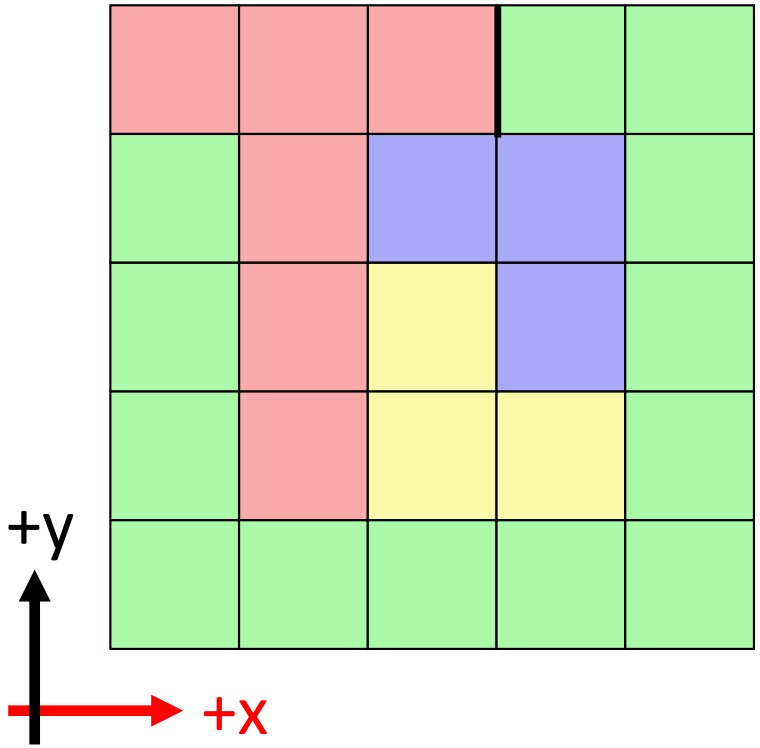


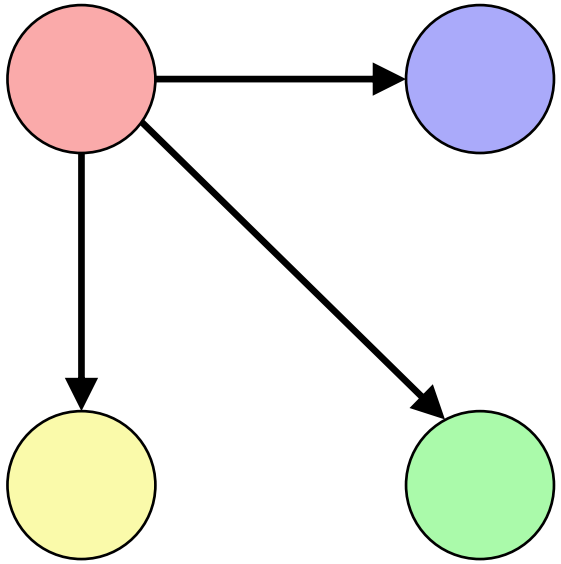
Test Interlocking: Directional Blocking Graph



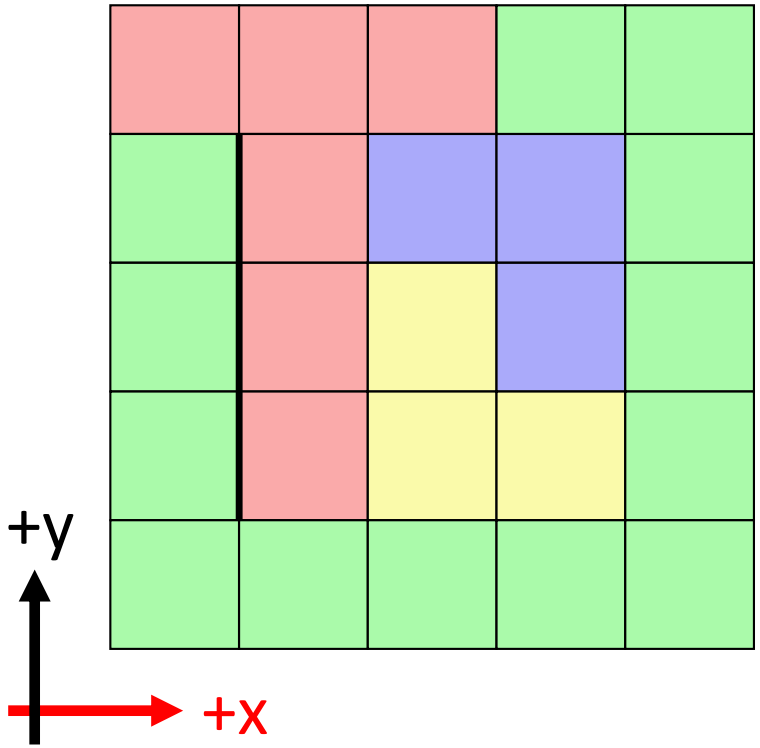


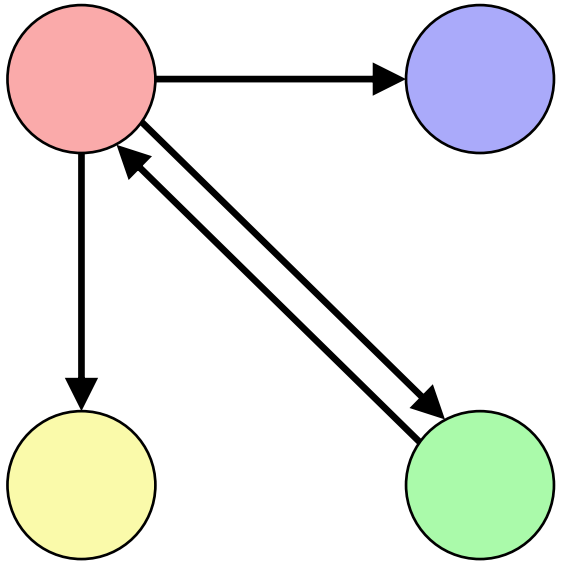
Test Interlocking: Directional Blocking Graph



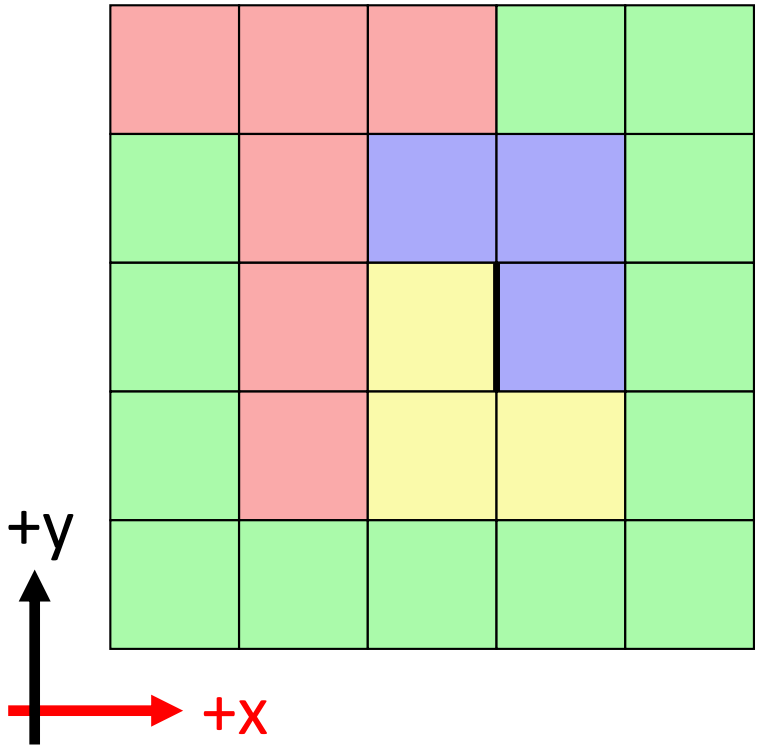


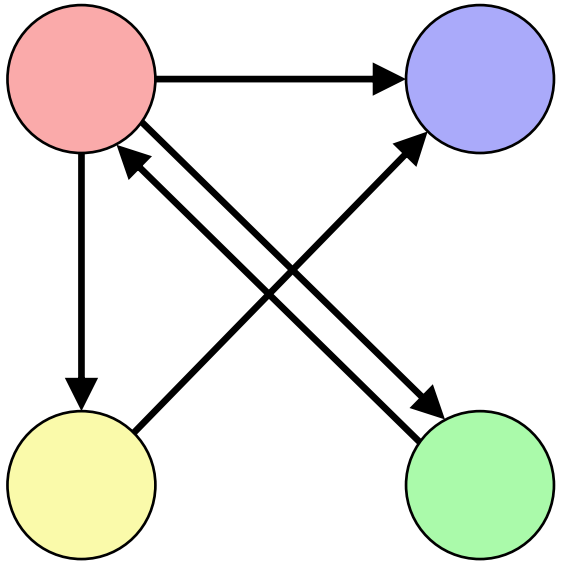
Test Interlocking: Directional Blocking Graph



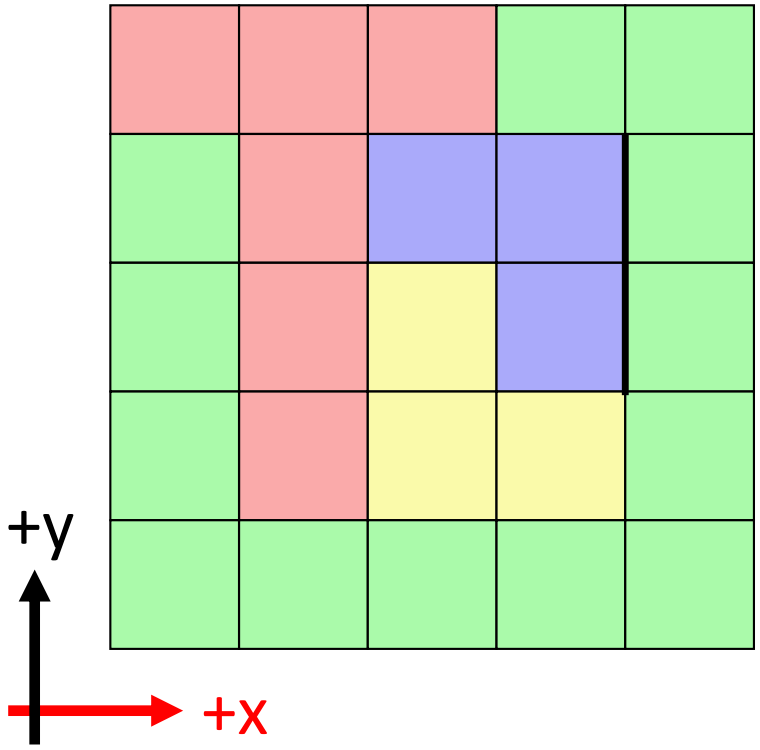


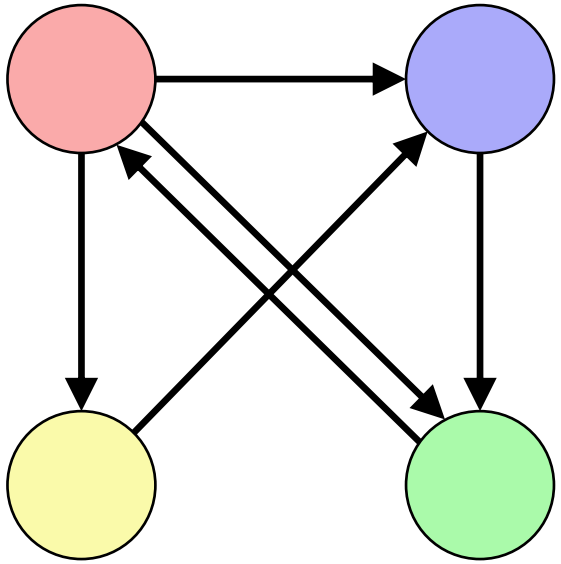
Test Interlocking: Directional Blocking Graph





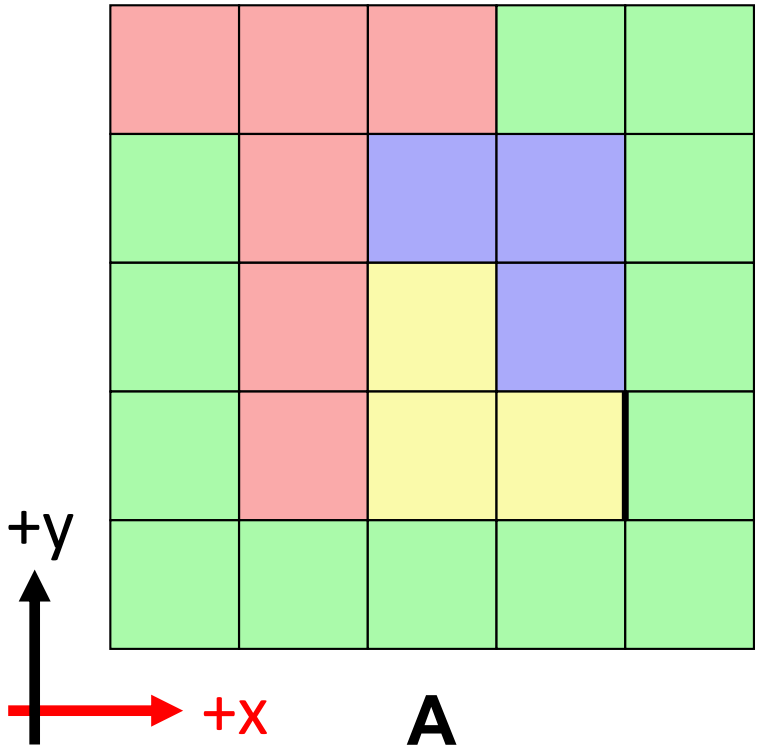
Test Interlocking: Directional Blocking Graph

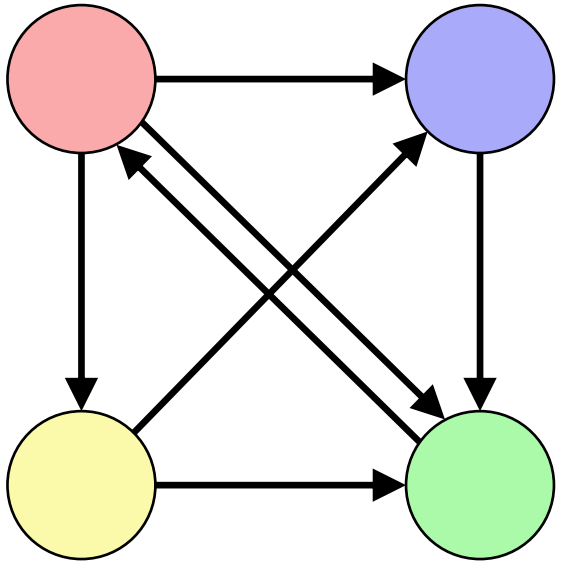




Test Interlocking: Directional Blocking Graph

Create a directional blocking graph for *d* = +x

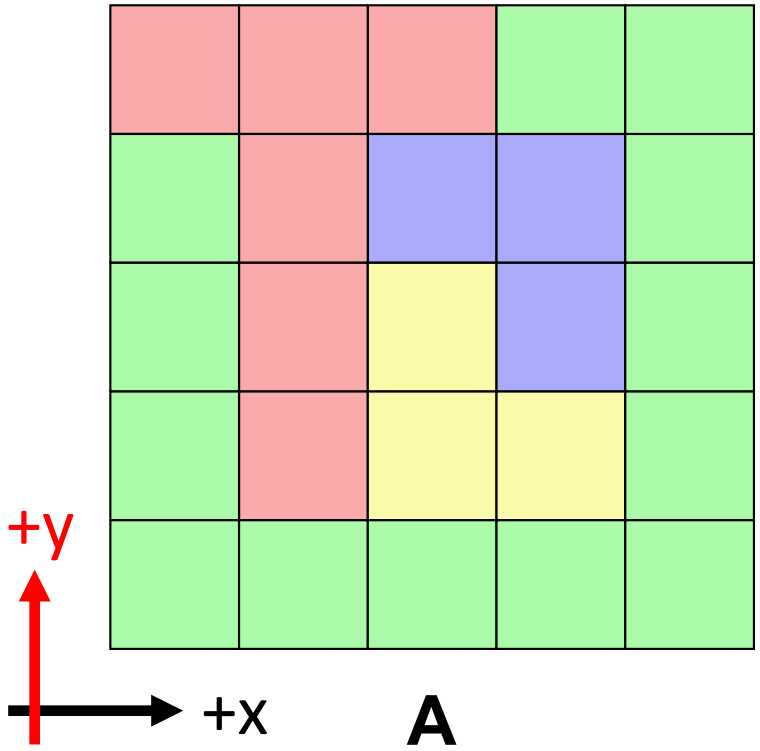




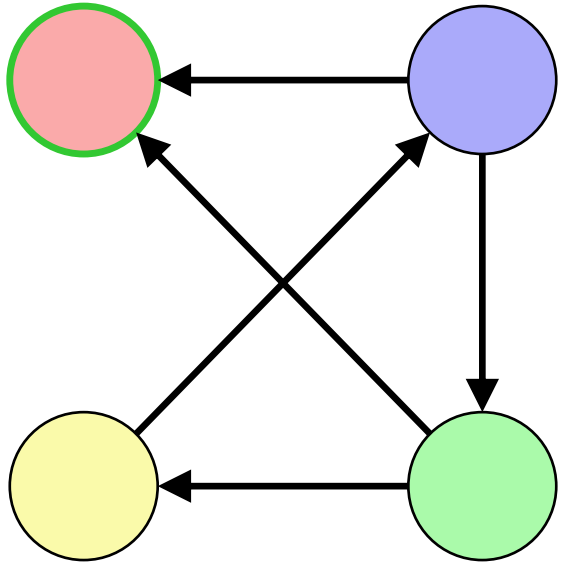
G(+x, **A**)

Test Interlocking: Directional Blocking Graph

Create a directional blocking graph for *d* = +y



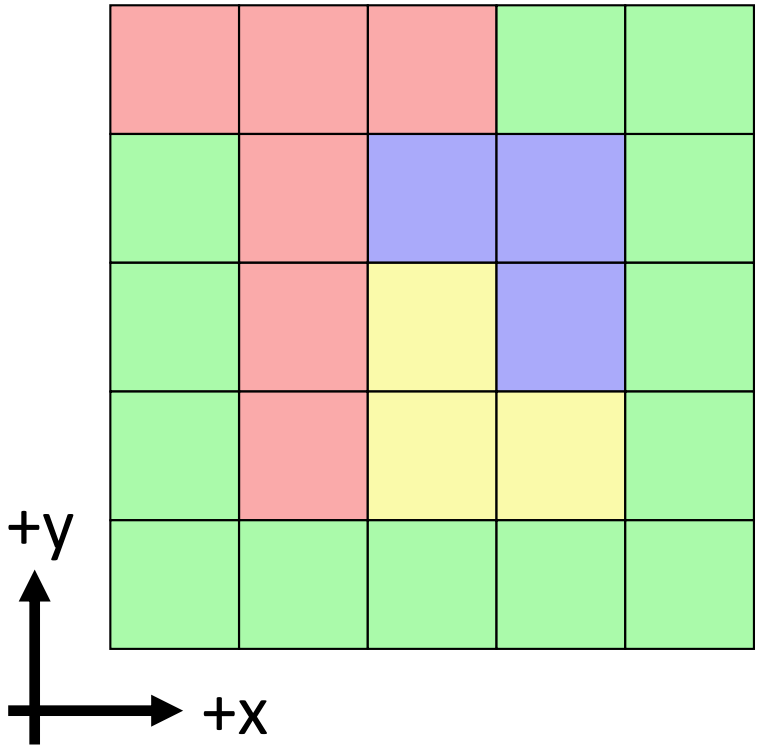
G(+x, **A**)

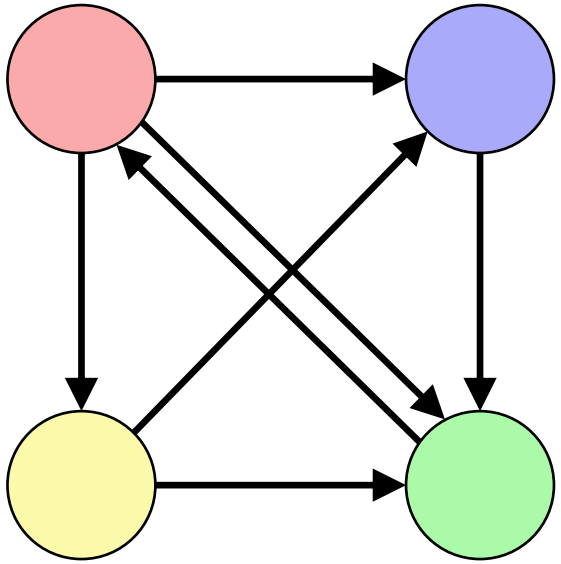


G(+y, **A**)

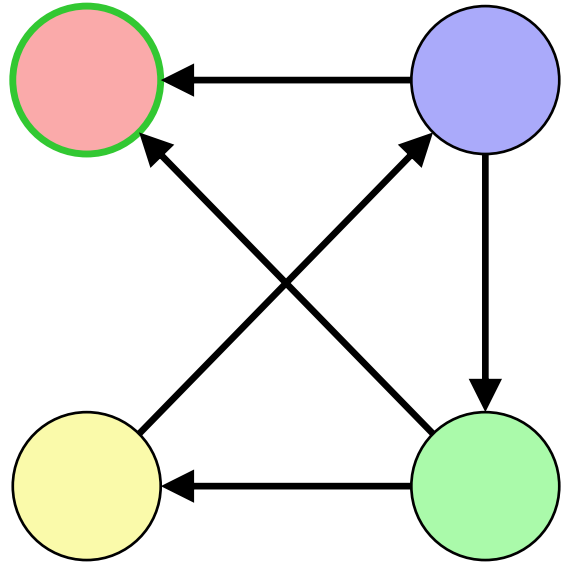
Test Interlocking: Directional Blocking Graph

The two graphs are called base directional blocking graphs of the assembly





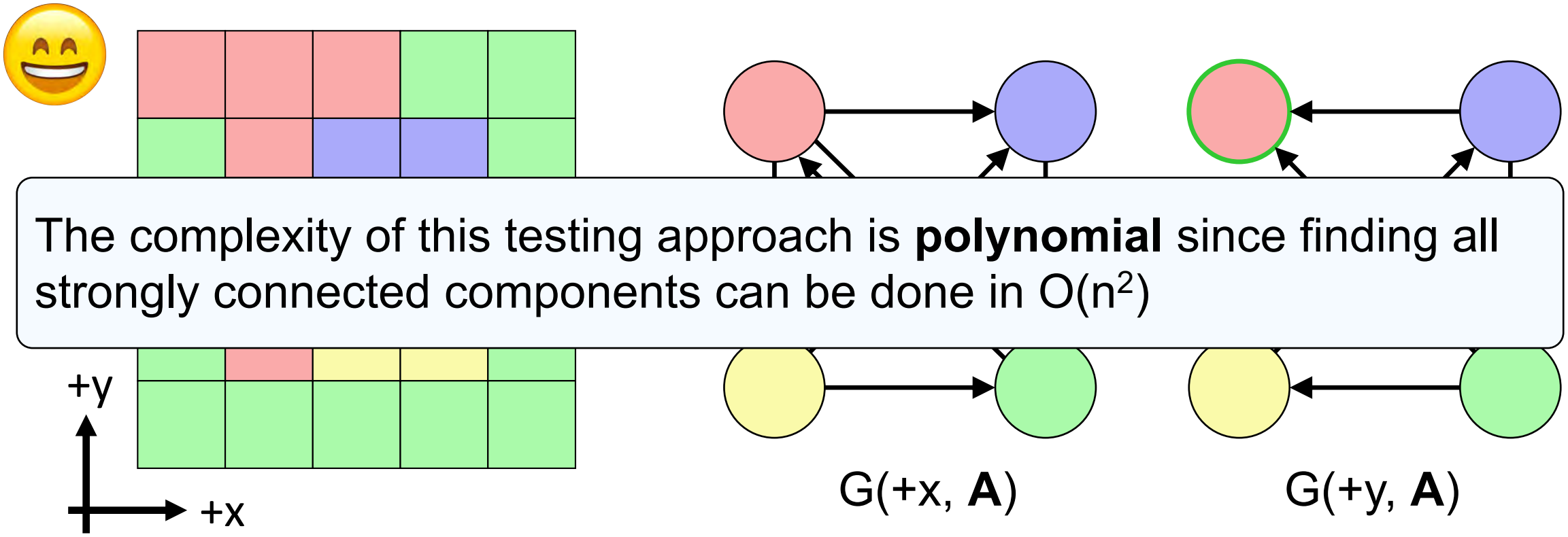
G(+x, **A**)



G(+y, **A**)

Test Interlocking: Directional Blocking Graph

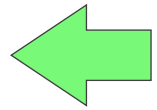
An assembly is interlocking if all base directional blocking graphs are strongly connected except the key.



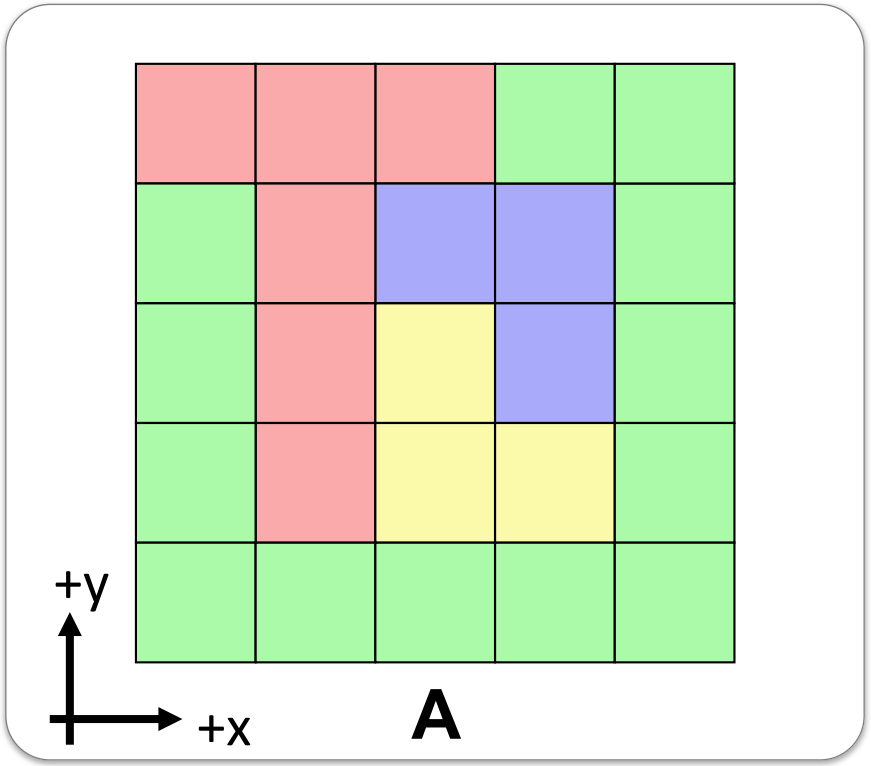
The complexity of this testing approach is **polynomial** since finding all strongly connected components can be done in $O(n^2)$

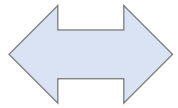
Contribution #2: Design Interlocking

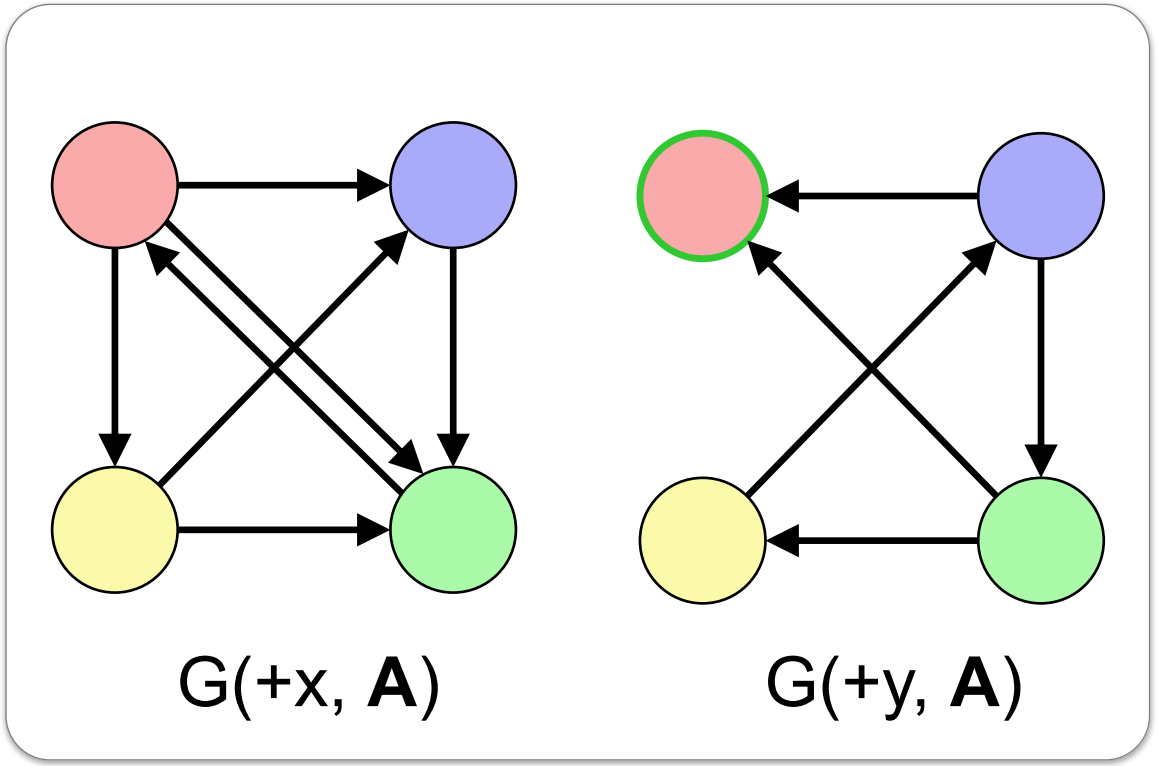
Construct interlocking parts geometry



Make all graphs strongly connected

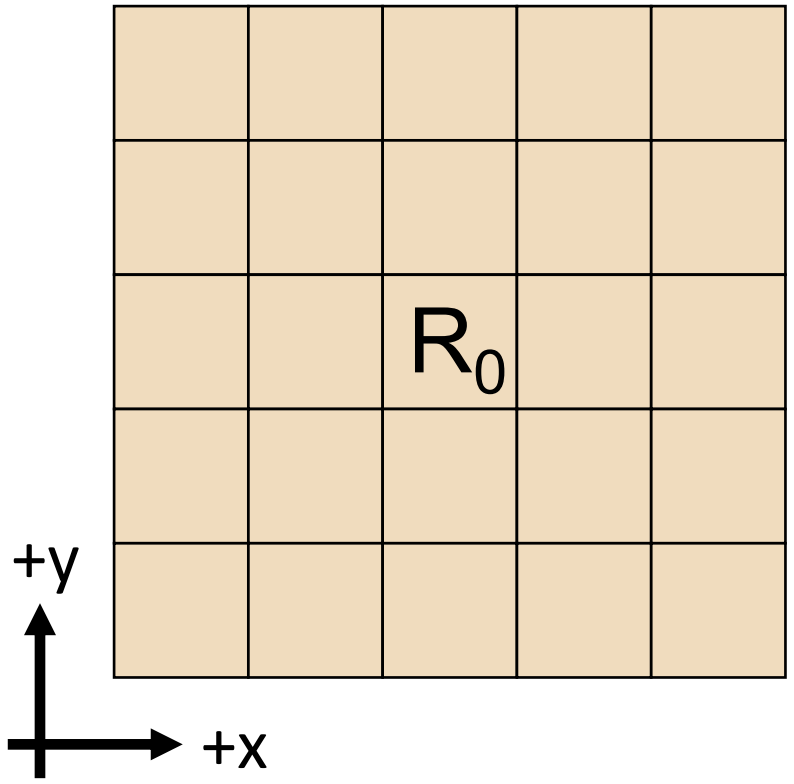






Design Interlocking: Iterative Construction

Geometry Space



Graph Space

G(+x, $\mathbf{A_0}$)

G(+y, $\mathbf{A_0}$)

Design Interlocking: Iterative Construction

Split nodes

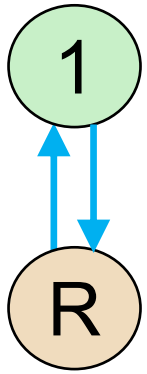
$G(+x, \mathbf{A_1})$

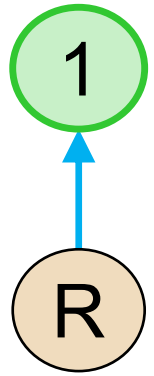
$G(+y, \mathbf{A_1})$

Design Interlocking: Iterative Construction

Graph Design

(make graphs strongly connected except the key)





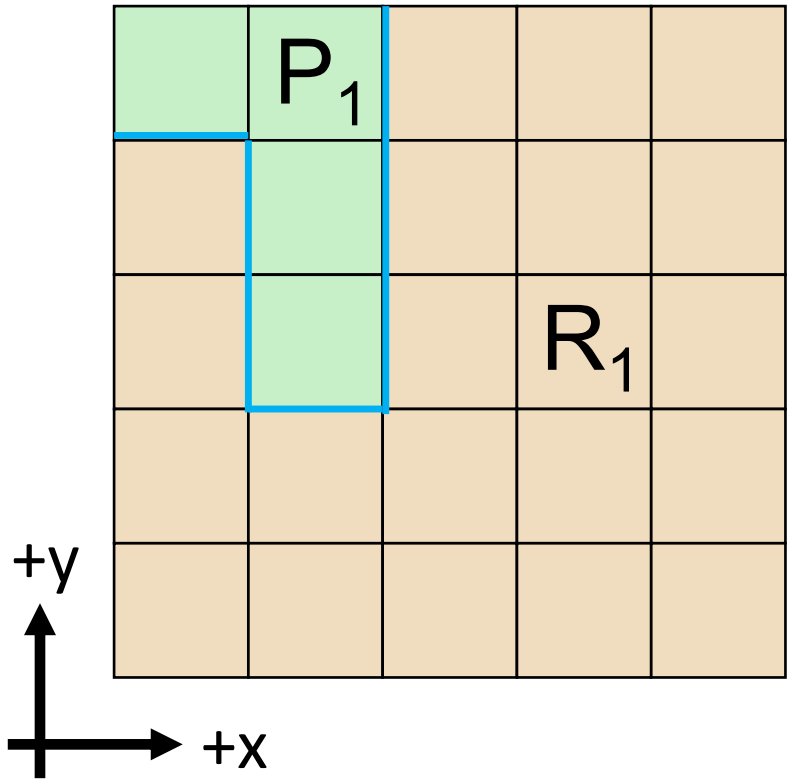
$G(+x, \mathbf{A_1})$

$G(+y, \mathbf{A_1})$

Design Interlocking: Iterative Construction

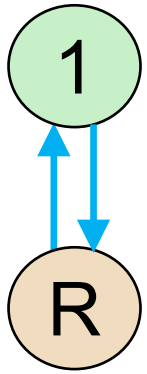
Geometry Realization

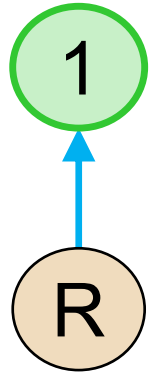
(realize blocking relations in the graphs)



Graph Design

(make graphs strongly connected except the key)



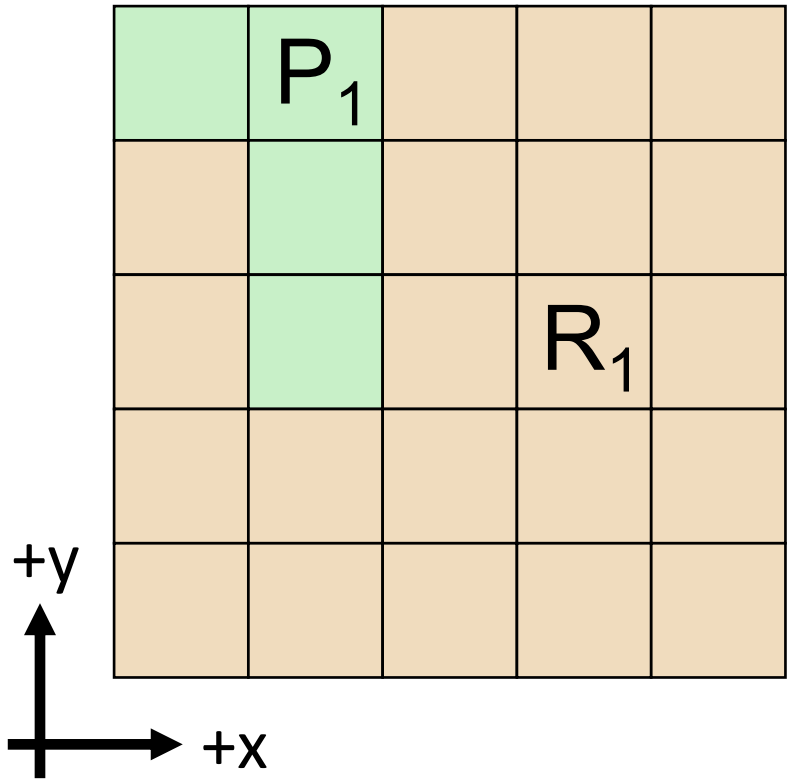


$G(+x, \mathbf{A_1})$

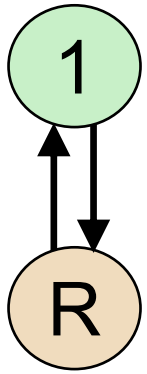
$G(+y, \mathbf{A_1})$

Design Interlocking: Iterative Construction

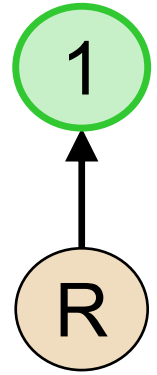
Geometry Realization



Graph Design



$G(+x, \mathbf{A_1})$

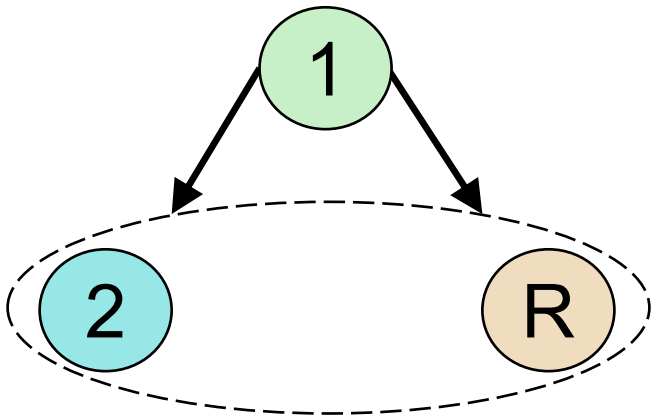


$G(+y, \mathbf{A_1})$

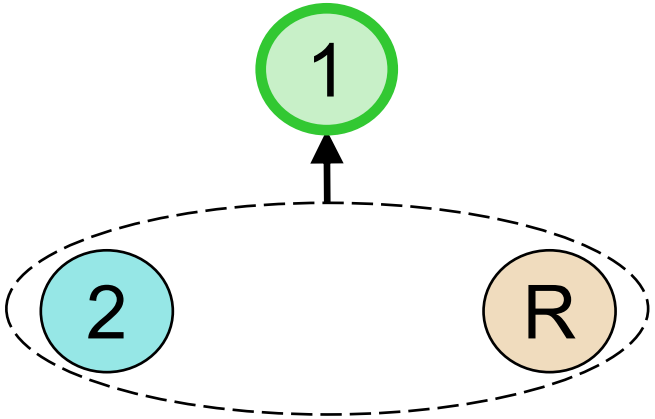
Design Interlocking: Iterative Construction

Geometry Realization

Graph Design



$G(+x, \mathbf{A_2})$

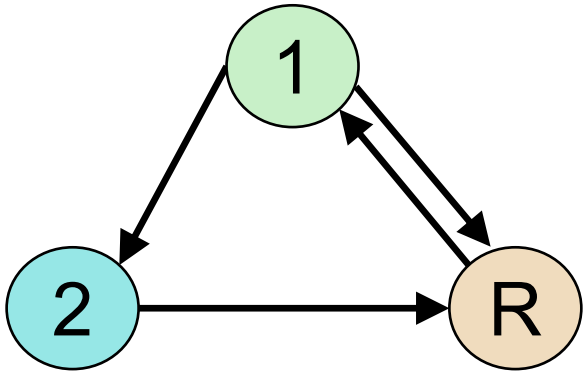


$G(+y, \mathbf{A_2})$

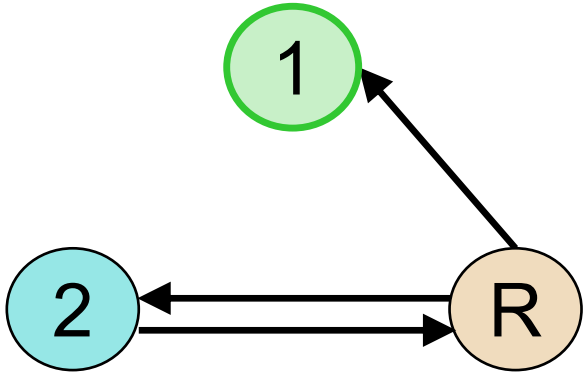
Design Interlocking: Iterative Construction

Geometry Realization

Graph Design



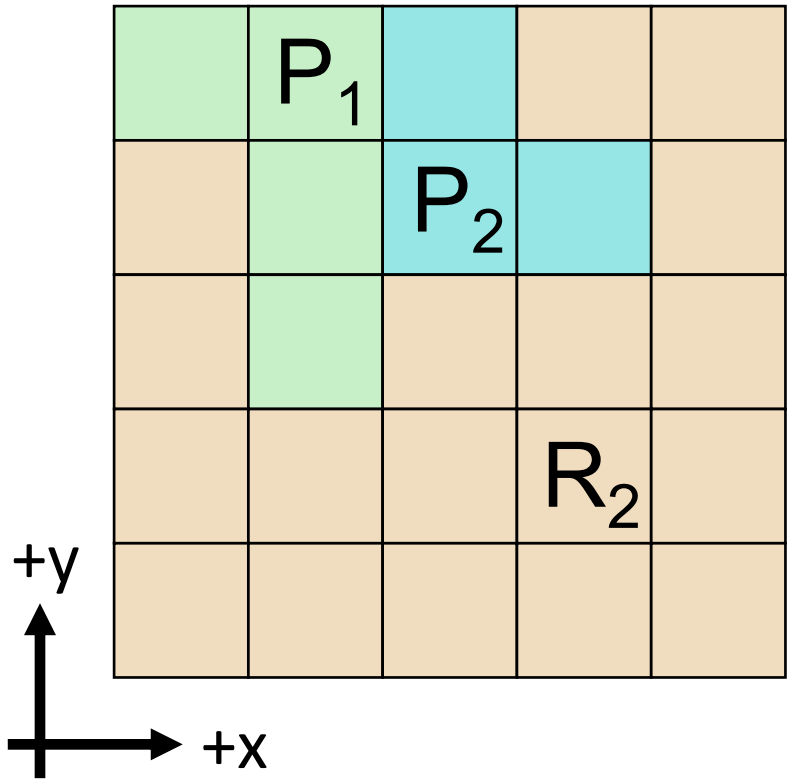
$G(+x, \mathbf{A_2})$



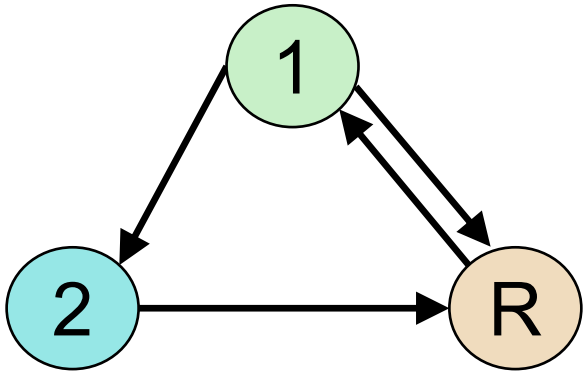
$G(+y, \mathbf{A_2})$

Design Interlocking: Iterative Construction

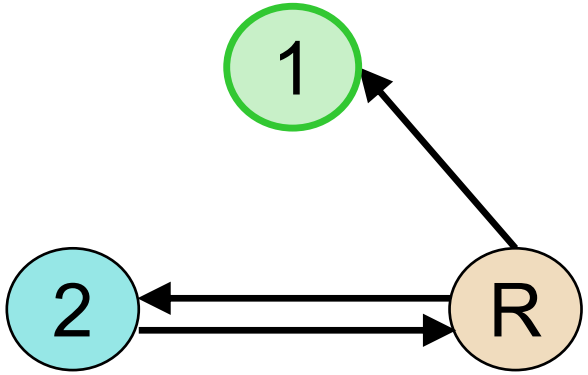
Geometry Realization



Graph Design



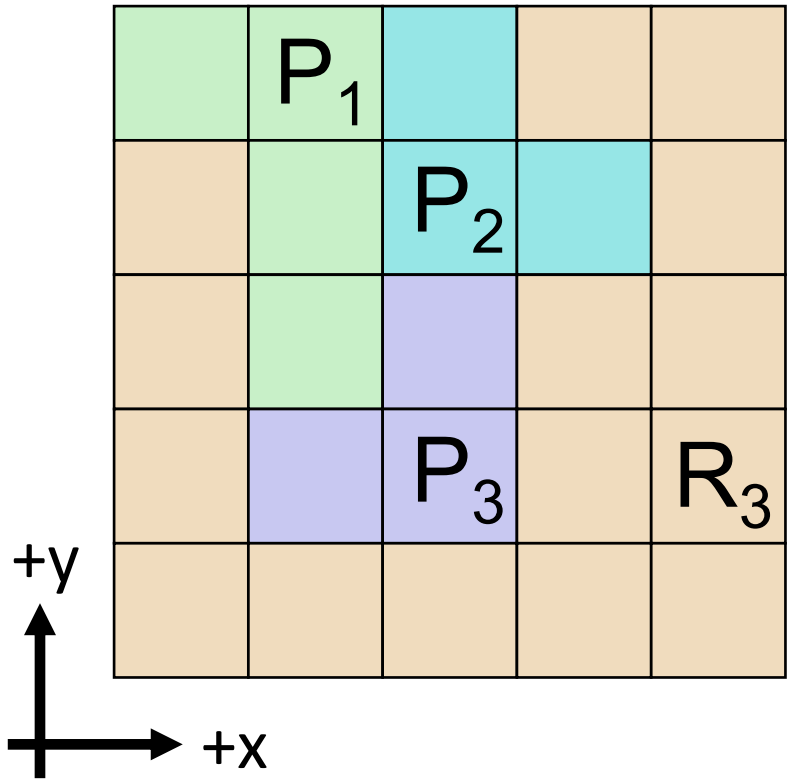
G(+x, $\mathbf{A_2}$)



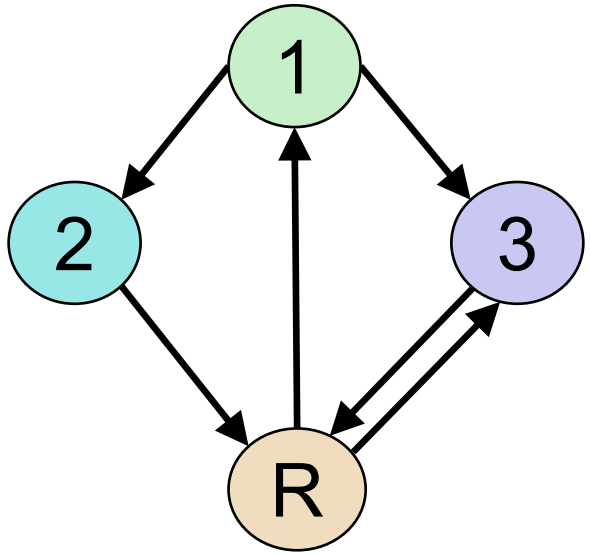
G(+y, $\mathbf{A_2}$)

Design Interlocking: Iterative Construction

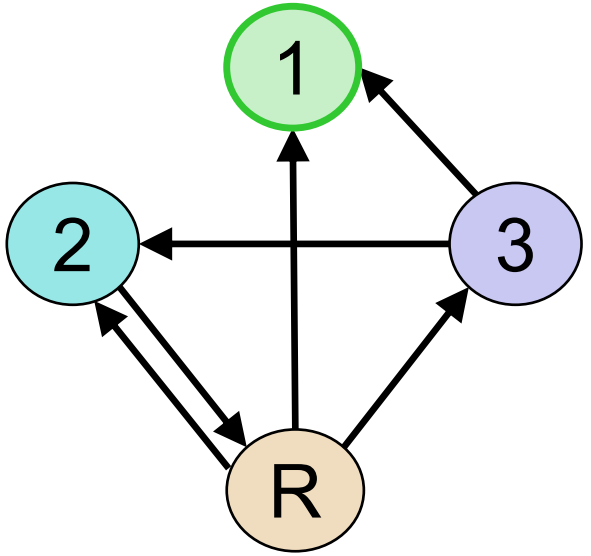
Geometry Realization



Graph Design



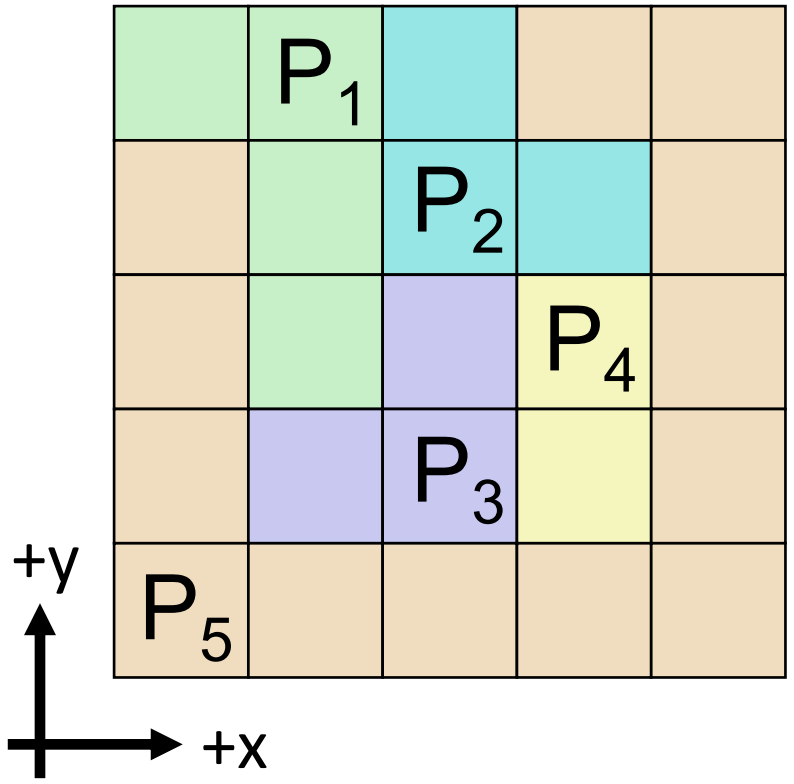
G(+x, $\mathbf{A_3}$)



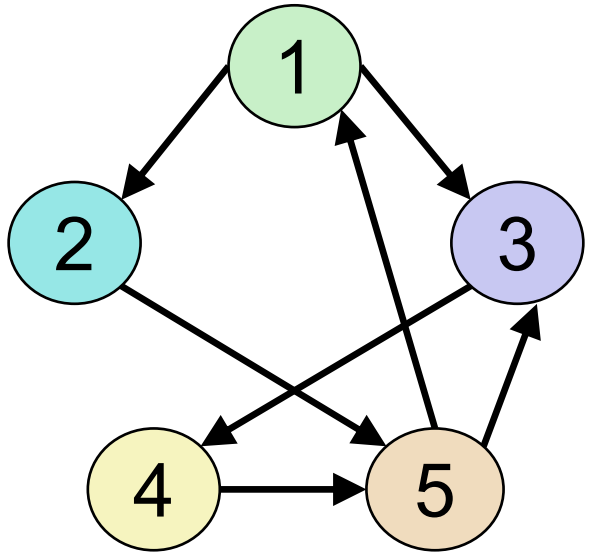
G(+y, $\mathbf{A_3}$)

Design Interlocking: Iterative Construction

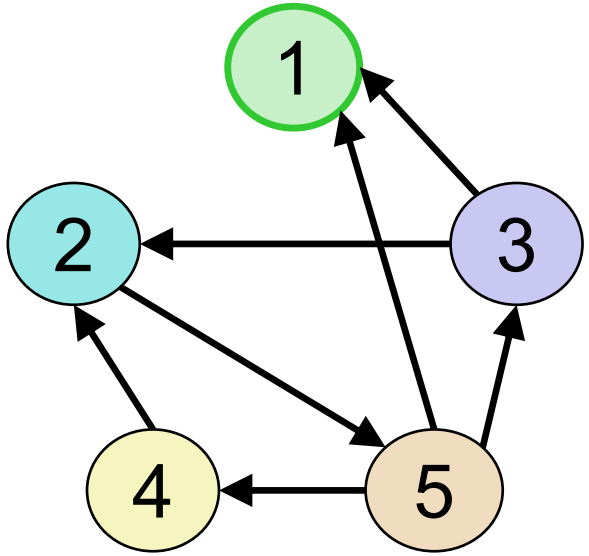
Geometry Realization



Graph Design



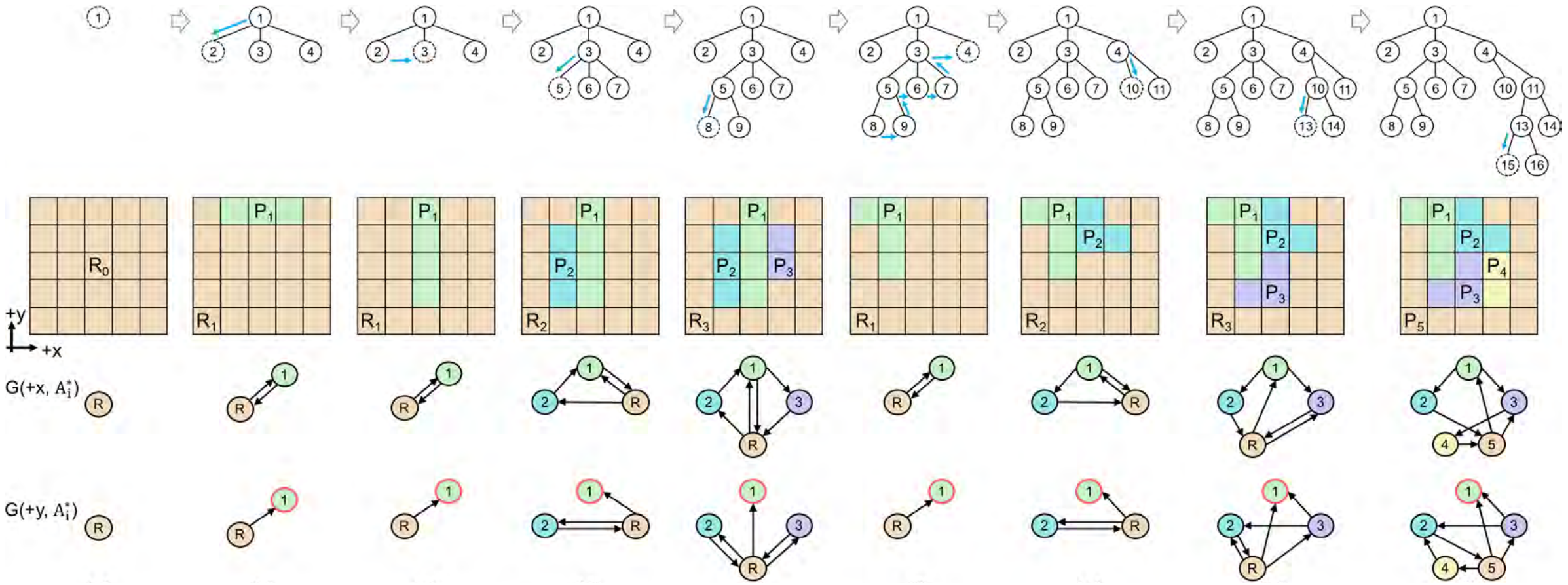
G(+x, $\mathbf{A_4}$)



G(+y, $\mathbf{A_4}$)

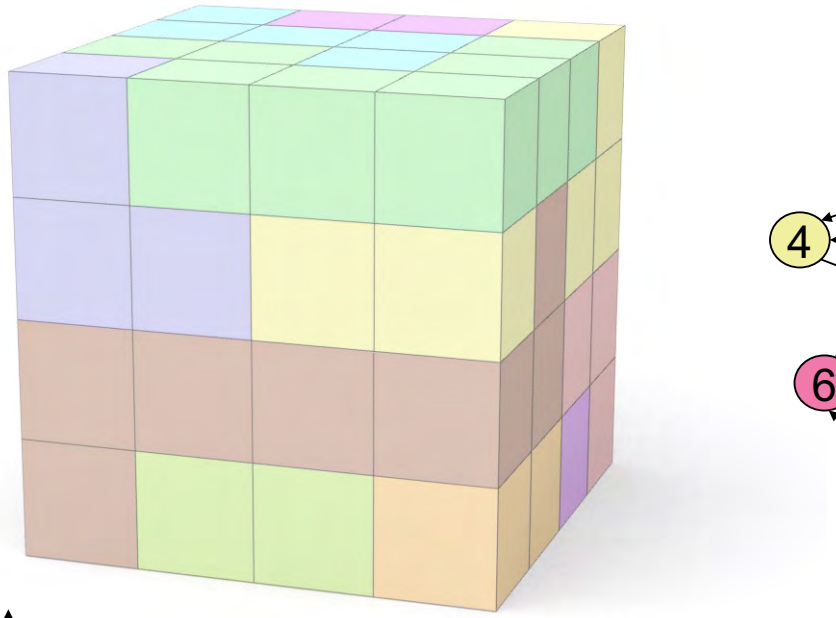
Design Interlocking: Tree-traversal Search

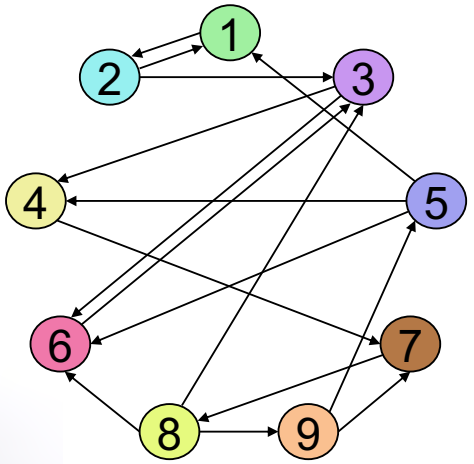
Search space is explored in a tree traversal process with automatic backtracking



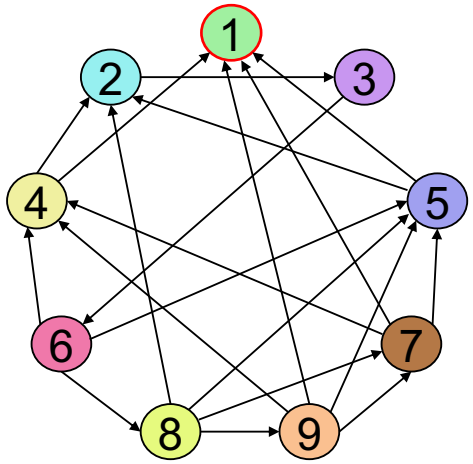
Results: Interlocking Voxelized Structures

9-part Interlocking **Cube**

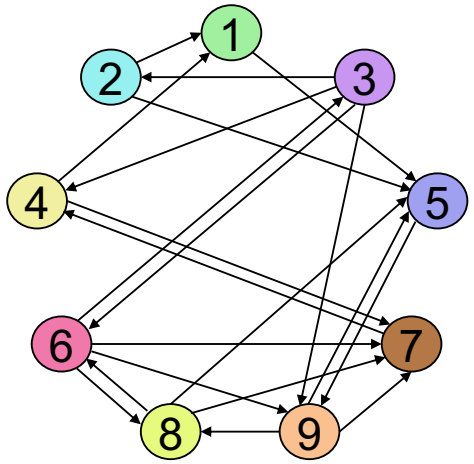




G(+x, A)



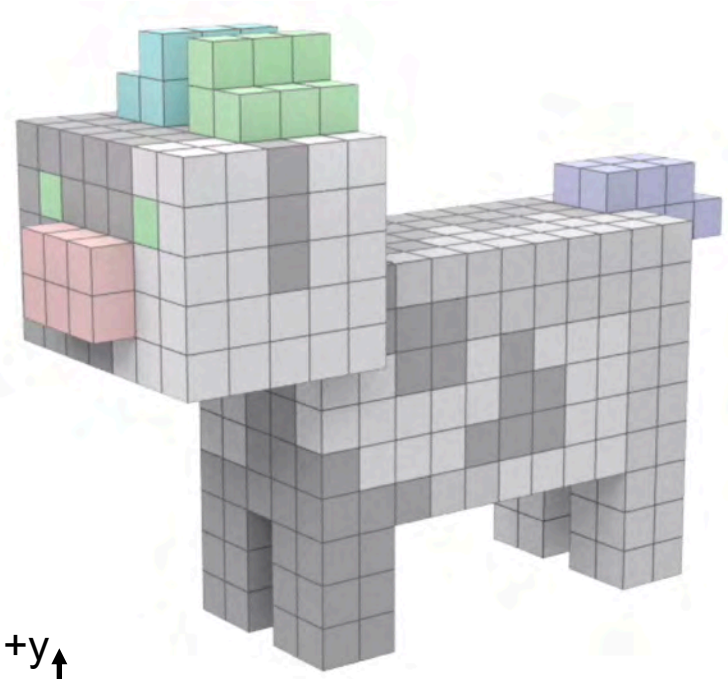
G(+y, A)

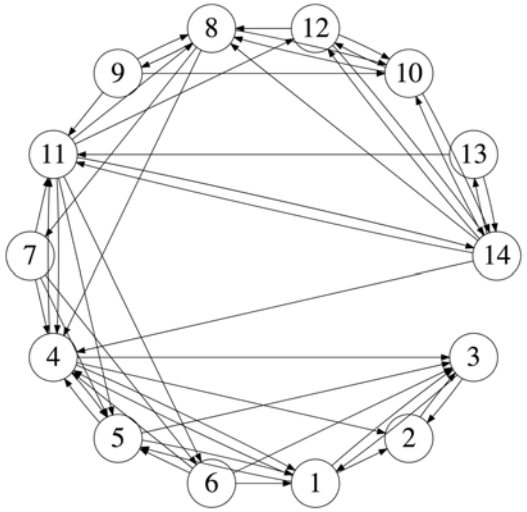


G(+z, A)

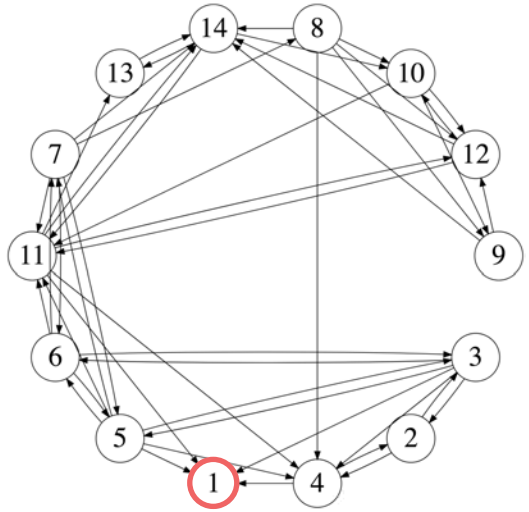
Results: Interlocking Voxelized Structures

14-part Interlocking **Dog**

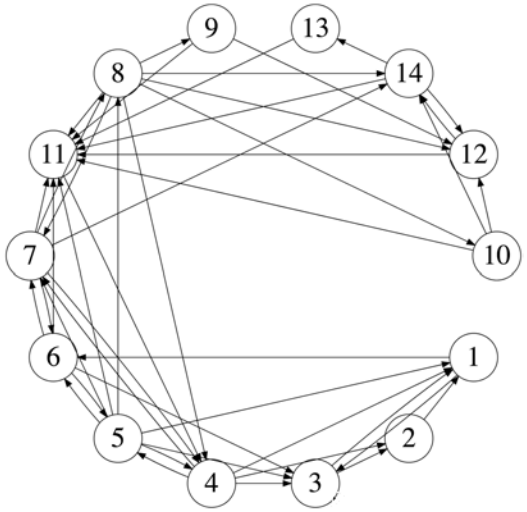




G(+x, A)



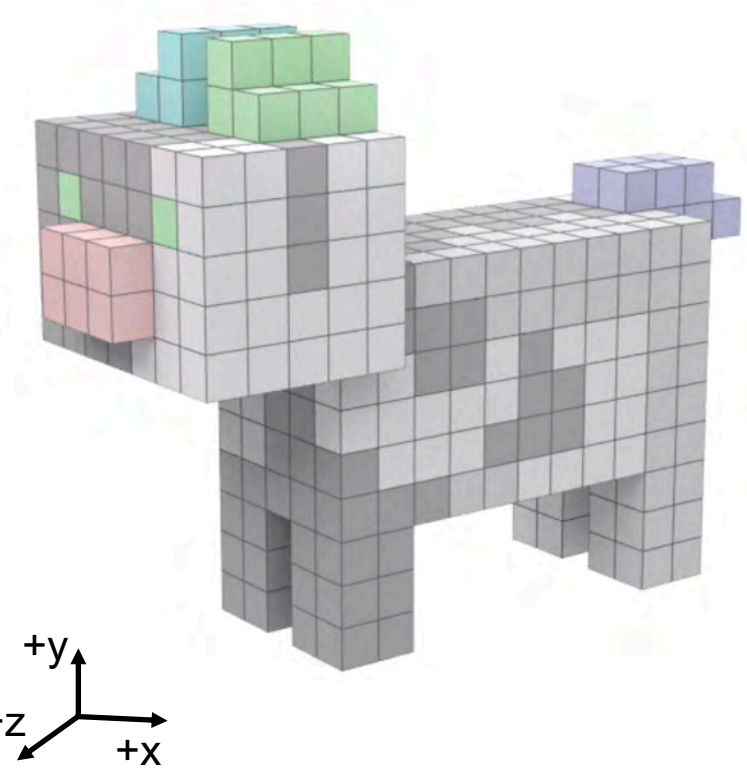
G(+y, A)



G(+z, A)

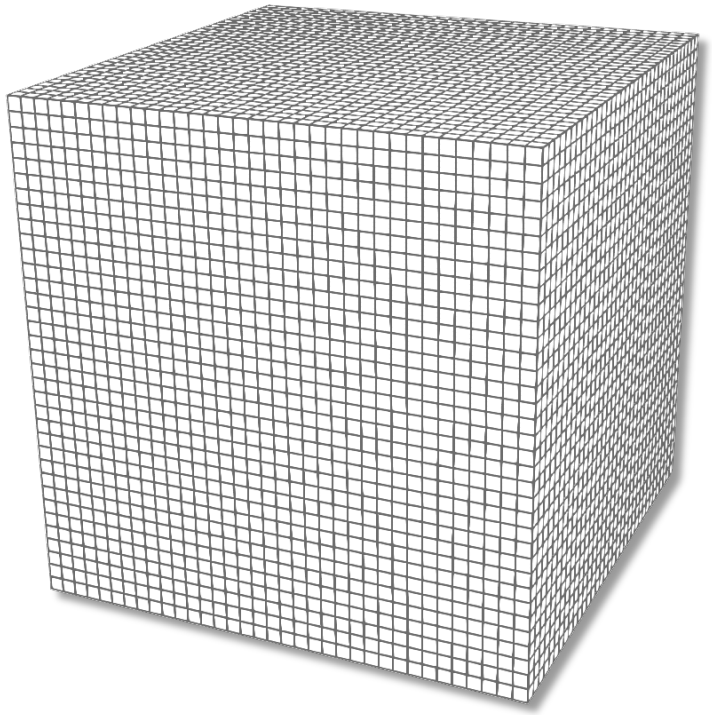
Results: Interlocking Voxelized Structures

14-part Interlocking **Dog**



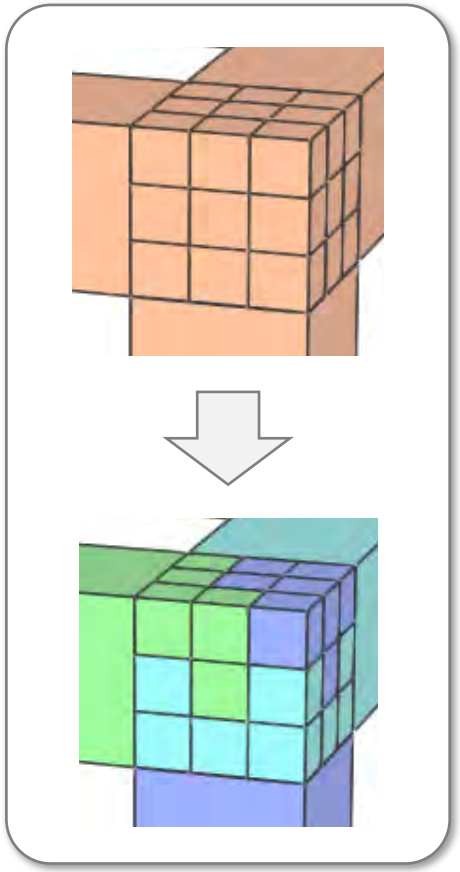
Results: Interlocking Voxelized Structures

1300-part Interlocking **Cube**

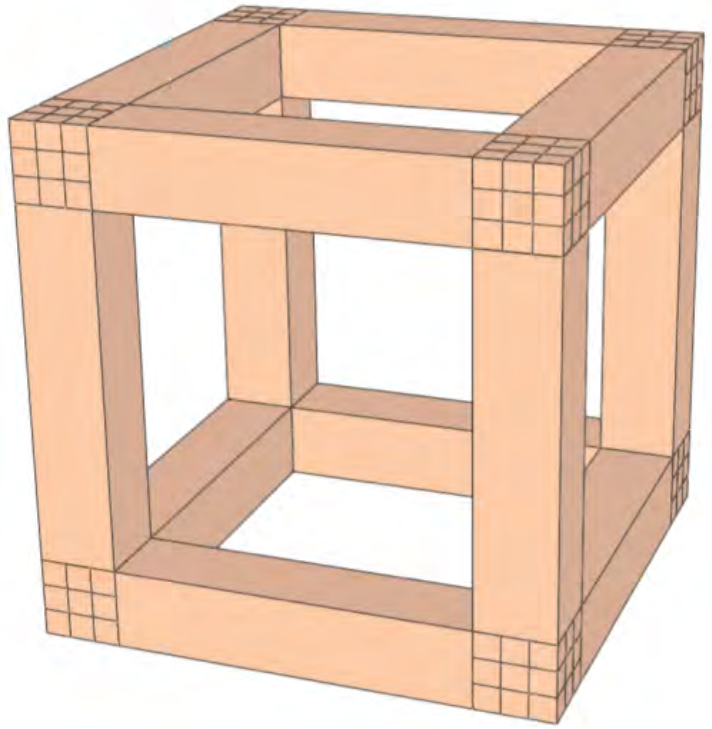


Input model

Results: Interlocking Frame Structures

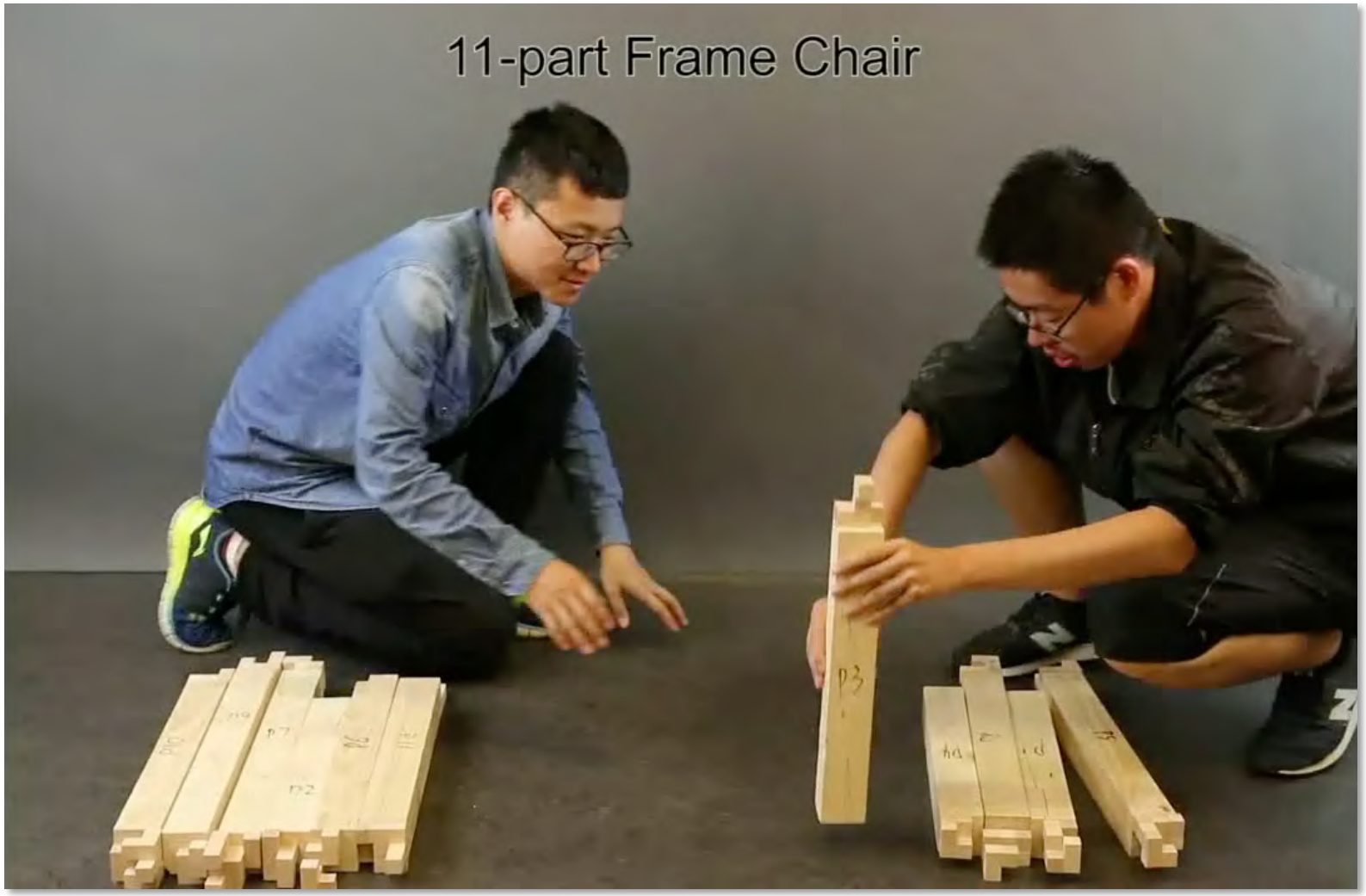


Voxel joints



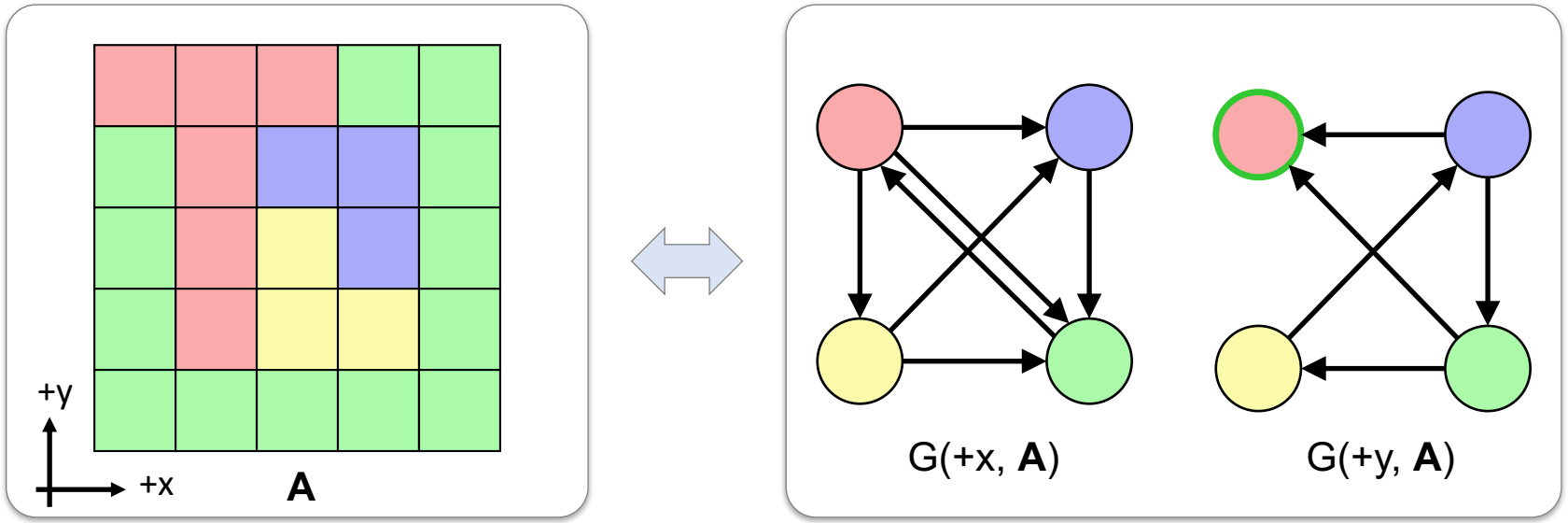
Results: Fabrication

Our designed Frame Chair



Summary of the project

- Make a **connection** between interlocking assemblies and a family of directional blocking graphs
- A graph-based method to **test** interlocking efficiently
- A graph-based method to **design** interlocking assemblies
  - graph design + geometry realization

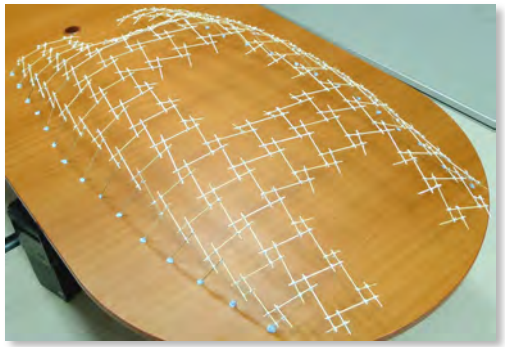


Outlook: Computational Design of Complex Assemblies

- Parts fabrication
- Parts joining
- Parts assembly
- Parts packing
- Structural Stability
- Reconfigurability
- Functionality






Thank You!

More information can be found at

https://sutd-cgl.github.io/

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

CGL