Synthesizing for Minimal Tile Sets
Patterned DNA Self-Assembly

Mika Göös & Pekka Orponen
Aalto University (School of Science and Technology)
Outline

1. Previous Study
2. Problem Definition
3. Approach of Ma & Lombardi
4. Our Contributions
Previous Study

Shapes modulo Scale

[Soloveichik & Winfree 2004]

Unsolvable

Gõös, Orponen (Aalto University) Synthesizing Minimal Tile Sets 15th June 2010
### Previous Study

<table>
<thead>
<tr>
<th>Shapes modulo Scale</th>
<th>Shapes</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Soloveichik &amp; Winfree 2004]</td>
<td>[Adleman et al. 2002]</td>
</tr>
</tbody>
</table>

| ππππ  | II II |

- **Unsolvable**
- **NP-hard**
Previous Study

**Shapes modulo Scale**
[Soloveichik & Winfree 2004]

**Shapes**
[Adleman et al. 2002]

**Patterns**
[Ma & Lombardi 2008]

- Unsolvable
- NP-hard
- Not known?
Pattern self-Assembly Tile set Synthesis (PATS)

Input

A $k$-colouring $c : [m] \times [n] \rightarrow [k]$
Pattern self-Assembly Tile set Synthesis (PATS)

**Input**

A $k$-colouring $c : [m] \times [n] \rightarrow [k]$

**Output**

A Tile Assembly System $\mathcal{F} = (T, S, s, 2)$
Pattern self-Assembly Tile set Synthesis (PATS)

Input
A $k$-colouring $c : [m] \times [n] \rightarrow [k]$

Output
A Tile Assembly System $\mathcal{T} = (T, S, s, 2)$
Pattern self-Assembly Tile set Synthesis (PATS)

**Input**

A $k$-colouring $c : [m] \times [n] \rightarrow [k]$ 

**Output**

A Tile Assembly System $\mathcal{T} = (T, S, s, 2)$
Pattern self-Assembly Tile set Synthesis (PATS)

Input
A $k$-colouring $c : \left[m \right] \times \left[n \right] \rightarrow \left[k \right]$

Output
A Tile Assembly System $\mathcal{T} = (T, S, s, 2)$
Pattern self-Assembly Tile set Synthesis (PATS)

**Input**

A $k$-colouring $c : [m] \times [n] \rightarrow [k]$

**Output**

A Tile Assembly System $\mathcal{T} = (T, S, s, 2)$
Pattern self-Assembly Tile set Synthesis (PATS)

Input
A $k$-colouring $c : [m] \times [n] \to [k]$

Output
A Tile Assembly System $\mathcal{F} = (T, S, s, 2)$
Input

A $k$-colouring $c : [m] \times [n] \rightarrow [k]$

Output

A Tile Assembly System $\mathcal{T} = (T, S, s, 2)$
Pattern self-Assembly Tile set Synthesis (PATS)

Input
A \( k \)-colouring \( c : [m] \times [n] \rightarrow [k] \)

Output
A Tile Assembly System \( \mathcal{F} = (T, S, s, 2) \)
Pattern self-Assembly Tile set Synthesis (PATS)

**Input**

A $k$-colouring $c : [m] \times [n] \rightarrow [k]$

**Output**

A Tile Assembly System $\mathcal{T} = (T, S, s, 2)$
Pattern self-Assembly Tile set Synthesis (PATS)

**Given:** A $k$-colouring $c : [m] \times [n] \rightarrow [k]$.

**Find:** A tile assembly system $\mathcal{T} = (T, S, s, 2)$ s.t.

- **P1.** The tiles in $T$ have bonding strength 1.
- **P2.** The domain of $S$ is $[0, m] \times \{0\} \cup \{0\} \times [0, n]$ and all the terminal assemblies have the domain $[0, m] \times [0, n]$.
- **P3.** There exists a colouring $d : T \rightarrow [k]$ such that for each terminal assembly $A \in \text{Term} \mathcal{T}$ we have $d(A(x, y)) = c(x, y)$ for all $(x, y) \in [m] \times [n]$. 
Approach of Ma & Lombardi

<table>
<thead>
<tr>
<th>3</th>
<th>0</th>
<th>1</th>
<th>4</th>
<th>5</th>
<th>7</th>
<th>8</th>
<th>10</th>
<th>11</th>
<th>13</th>
<th>14</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>21</td>
<td>22</td>
<td>32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>19</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>23</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td></td>
<td></td>
<td>36</td>
<td>38</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td></td>
<td></td>
<td>36</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td></td>
<td></td>
<td></td>
<td>40</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td></td>
<td></td>
<td></td>
<td>40</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>46</td>
<td></td>
<td></td>
<td></td>
<td>51</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>46</td>
<td></td>
<td></td>
<td></td>
<td>51</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>46</td>
<td></td>
<td>61</td>
<td>64</td>
<td>66</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>46</td>
<td></td>
<td>61</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>59</td>
<td></td>
<td>62</td>
<td>64</td>
<td>66</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>59</td>
<td></td>
<td>62</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>59</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

To minimize Tile set size:

- Merge glues
- Merge tiles

If conflicts arise:

- Continue merging!
Approach of Ma & Lombardi

|   | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 |
To minimize Tile set size:
- Merge glues
- Merge tiles

If conflicts arise:
- Continue merging!
Our Contributions

We present

1. Extension of the work of Ma & Lombardi
2. Branch & Bound algorithm
3. Pruning heuristics
Lemma: Minimal solutions to the PATS problem are deterministic.
**Lemma:** Minimal solutions to the PATS problem are *deterministic*.
Partition Centric View

A constructible partition of $[m] \times [n]$ is coarser than...
Partition Centric View

A constructible partition of $[m] \times [n]$ is coarser than $G$.
Partition Centric View

**Contructible** partition of $[m] \times [n]$
Partition Centric View

is coarser than
Searching the Lattice of Partitions

\[ \begin{align*}
\text{Constructible partition} & \quad \text{Our B&B algorithm} \\
\text{Node-disjoint search tree} & \quad \text{Uses memory } \text{poly} (mn) \\
\text{Branching only on constructible partitions} & \\
\text{Cheap bounding function} & 
\end{align*} \]
Searching the Lattice of Partitions

- Constructible partition
- Our B&B algorithm
- Node-disjoint search tree
- Uses memory $\text{poly}(mn)$
- Branching only on constructible partitions
- Cheap bounding function

Göös, Orponen (Aalto University)
Searching the Lattice of Partitions

- Contructible partition
- Our B&B algorithm
- Node-disjoint search tree
- Uses memory poly$(mn)$
- Branching only on constructible partitions
- Cheap bounding function

Göös, Orponen (Aalto University)
Synthesizing Minimal Tile Sets
15th June 2010 10 / 13
Searching the Lattice of Partitions

Our B&B algorithm

- Node-disjoint search tree
- Uses memory poly\((mn)\)
- Branching only on constructible partitions
- Cheap bounding function

Göös, Orponen (Aalto University)
Searching the Lattice of Partitions

- Our B&B algorithm

1. Node-disjoint search tree
2. Uses memory \( \text{poly}(mn) \)
3. Branching only on constructible partitions
4. Cheap bounding function

Contructible partition

Synthesizing Minimal Tile Sets
Göös, Orponen (Aalto University)
15th June 2010
Contructible partition

Our 

Node-disjoint search tree

Uses memory $\text{poly}(mn)$

Branching only on constructible partitions

Cheap bounding function
Searching the Lattice of Partitions

Our B&B algorithm

1. Node-disjoint search tree
2. Uses memory poly(mn)
3. Branching only on constructible partitions
4. Cheap bounding function

Contructible partition

Göös, Orponen (Aalto University) Synthesizing Minimal Tile Sets 15th June 2010 10 / 13
Searching the Lattice of Partitions

Contructible partition

Our B&B algorithm

1. Node-disjoint search tree
2. Uses memory poly(mn)
3. Branching only on constructible partitions
4. Cheap bounding function

Göös, Orponen (Aalto University)
Synthesizing Minimal Tile Sets
15th June 2010 10 / 13
Searching the Lattice of Partitions

Our B&B algorithm

1. Node-disjoint search tree
2. Uses memory $\text{poly}(mn)$
3. Branching only on constructible partitions
4. Cheap bounding function

Contructible partition
Searching the Lattice of Partitions

Contructible partition

Our B&B algorithm

1. Node-disjoint search tree
2. Uses memory poly(mn)
3. Branching only on constructible partitions
4. Cheap bounding function
Searching the Lattice of Partitions

Our B&B algorithm

1. Node-disjoint search tree
2. Uses memory poly\((mn)\)
3. Branching only on constructible partitions
4. Cheap bounding function
Contructible partition

Our B&B algorithm

1. Node-disjoint search tree
2. Uses memory poly(mn)
3. Branching only on constructible partitions
4. Cheap bounding function
Searching the Lattice of Partitions

Our B&B algorithm

1. Node-disjoint search tree
2. Uses memory $\text{poly}(mn)$
3. Branching only on constructible partitions
Searching the Lattice of Partitions

Our B&B algorithm

1. Node-disjoint search tree
2. Uses memory $\text{poly}(mn)$
3. Branching only on constructible partitions
4. Cheap bounding function

$mn$

$mn-1$

$mn-2$

3

2

1
Running time on random 2-coloured instances

$\sim 2^{mn}$
Conclusions

PATS problem remains challenging

- Open Problems:
  1. Is it $\text{NP}$-hard?
  2. Faster algorithms?
  3. Generalize to infinite finite-period patterns

- PATS is of practical importance
Thank you!