Computational Complexity and Bounds for Norinori and LITS

Michael Biro, Christiane Schmidt
Norinori? LITS?
Norinori? LITS?
• Pencil-and-paper puzzles
Norinori? LITS?

- Pencil-and-paper puzzles
- Made popular by Japanese publisher Nikoli
Norinori? LITS?
• Pencil-and-paper puzzles
• Made popular by Japanese publisher Nikoli
• (Norinori = Dominnocuous, LITS = Nuruomino)
Norinori? LITS?

- Pencil-and-paper puzzles
- Made popular by Japanese publisher Nikoli
- (Norinori = Dominnocuous, LITS = Nuruomino)
- Both played on mxn square grid partitioned into connected polyomino regions
Norinori? LITS?

- Pencil-and-paper puzzles
- Made popular by Japanese publisher Nikoli
- (Norinori = Dominnocuous, LITS = Nuruomino)
- Both played on mxn square grid partitioned into connected polyomino regions
- Place black squares in the polyomino regions
Norinori
Place black squares in the polyominoes, such that the final board satisfies
Place black squares in the polyominoes, such that the final board satisfies
• Each black square has exactly one black neighbour.
Place black squares in the polyominoes, such that the final board satisfies
• Each black square has exactly one black neighbour.
• There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies
- Each black square has exactly one black neighbour.
- There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies
• Each black square has exactly one black neighbour.
• There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies
• Each black square has exactly one black neighbour.
• There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies
• Each black square has exactly one black neighbour.
• There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies
• Each black square has exactly one black neighbour.
• There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies:

- Each black square has exactly one black neighbour.
- There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies

- Each black square has exactly one black neighbour.
- There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies
• Each black square has exactly one black neighbour.
• There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies

- Each black square has exactly one black neighbour.
- There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies
• Each black square has exactly one black neighbour.
• There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies

- Each black square has exactly one black neighbour.
- There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies

- Each black square has exactly one black neighbour.
- There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies:

- Each black square has exactly one black neighbour.
- There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies
• Each black square has exactly one black neighbour.
• There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies
• Each black square has exactly one black neighbour.
• There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies

- Each black square has exactly one black neighbour.
- There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies

- Each black square has exactly one black neighbour.
- There are exactly 2 black squares in each polyomino region.
Place black squares in the polyominoes, such that the final board satisfies

- Each black square has exactly one black neighbour.
- There are exactly 2 black squares in each polyomino region.
Theorem 1:
Determining if a Norinori board is solvable is NP-complete and counting the number of solutions is #P-complete.
Theorem 1:
Determining if a Norinori board is solvable is NP-complete and counting the number of solutions is #P-complete.

Proof by reduction from PLANAR 1-IN-3-SAT:
Theorem 1:
Determining if a Norinori board is solvable is NP-complete and counting the number of solutions is \#P-complete.

Proof by reduction from PLANAR 1-IN-3-SAT:
- G: incidence graph of an instance F of PLANAR 1-IN-3-SAT
Theorem 1:
Determining if a Norinori board is solvable is NP-complete and counting the number of solutions is #P-complete.

Proof by reduction from PLANAR 1-IN-3-SAT:
- G: incidence graph of an instance F of PLANAR 1-IN-3-SAT
- We use rectilinear embedding of G and turn it into Norinori board B
Theorem 1:
Determining if a Norinori board is solvable is NP-complete and counting the number of solutions is #P-complete.

Proof by reduction from PLANAR 1-IN-3-SAT:
• G: incidence graph of an instance F of PLANAR 1-IN-3-SAT
• We use rectilinear embedding of G and turn it into Norinori board B
• Solution to B yields a solution to F (NP-hardness)
Theorem 1:
Determining if a Norinori board is solvable is NP-complete and counting the number of solutions is \#P-complete.

Proof by reduction from PLANAR 1-IN-3-SAT:
• G: incidence graph of an instance F of PLANAR 1-IN-3-SAT
• We use rectilinear embedding of G and turn it into Norinori board B
• Solution to B yields a solution to F (\implies \text{NP-hardness})
• Given a solution to an mxn Norinori board, it can be verified in polynomial time
Theorem 1:
Determining if a Norinori board is solvable is NP-complete and counting the number of solutions is #P-complete.

Proof by reduction from PLANAR 1-IN-3-SAT:
• G: incidence graph of an instance F of PLANAR 1-IN-3-SAT
• We use rectilinear embedding of G and turn it into Norinori board B
• Solution to B yields a solution to F (NP-hardness)
• Given a solution to an mxn Norinori board, it can be verified in polynomial time
• One-to-one correspondence between solutions of B and solutions of F (#P-complete)
Theorem 1:
Determining if a Norinori board is solvable is NP-complete and counting the number of solutions is \#P-complete.

Proof by reduction from PLANAR 1-IN-3-SAT:

• G: incidence graph of an instance F of PLANAR 1-IN-3-SAT
• We use rectilinear embedding of G and turn it into Norinori board B
• Solution to B yields a solution to F (\(\Rightarrow\) NP-hardness)
• Given a solution to an mxn Norinori board, it can be verified in polynomial time
• One-to-one correspondence between solutions of B and solutions of F (\(\Rightarrow\)#P-complete)

Variable loop:
Theorem 1:
Determining if a Norinori board is solvable is NP-complete and counting the number of solutions is #P-complete.

Proof by reduction from PLANAR 1-IN-3-SAT:
- G: incidence graph of an instance F of PLANAR 1-IN-3-SAT
- We use rectilinear embedding of G and turn it into Norinori board B
- Solution to B yields a solution to F (⇒ NP-hardness)
- Given a solution to an mxn Norinori board, it can be verified in polynomial time
- One-to-one correspondence between solutions of B and solutions of F (⇒#P-complete)

Variable loop:

“false”
**Theorem 1:**
Determining if a Norinori board is solvable is NP-complete and counting the number of solutions is #P-complete.

Proof by reduction from PLANAR 1-IN-3-SAT:
• G: incidence graph of an instance F of PLANAR 1-IN-3-SAT
• We use rectilinear embedding of G and turn it into Norinori board B
• Solution to B yields a solution to F (\(\Rightarrow\) NP-hardness)
• Given a solution to an mxn Norinori board, it can be verified in polynomial time
• One-to-one correspondence between solutions of B and solutions of F (\(\Rightarrow\)#P-complete)

**Variable loop:**

```
false
```

```
true
```
**Theorem 1:**
Determining if a Norinori board is solvable is NP-complete and counting the number of solutions is #P-complete.

Proof by reduction from PLANAR 1-IN-3-SAT:
- G: incidence graph of an instance F of PLANAR 1-IN-3-SAT
- We use rectilinear embedding of G and turn it into Norinori board B
- Solution to B yields a solution to F (\(\Rightarrow\) NP-hardness)
- Given a solution to an mxn Norinori board, it can be verified in polynomial time
- One-to-one correspondence between solutions of B and solutions of F (\(\Rightarrow\)#P-complete)

**Variable loop:**

- "false"
- "true"

**Fixes squares in center face, and makes third solution to the loop infeasible**
Theorem 1:
Determining if a Norinori board is solvable is NP-complete and counting the number of solutions is \#P-complete.

Proof by reduction from PLANAR 1-IN-3-SAT:
• G: incidence graph of an instance F of PLANAR 1-IN-3-SAT
• We use rectilinear embedding of G and turn it into Norinori board B
• Solution to B yields a solution to F (\textarrow NP-hardness)
• Given a solution to an mxn Norinori board, it can be verified in polynomial time
• One-to-one correspondence between solutions of B and solutions of F (\textarrow #P-complete)

Variable loop:

Fixes squares in center face, and makes third solution to the loop infeasible

“false”

“true”
Face gadget, for any open region:
**Face gadget**, for any open region:

**Corridor gadget**, propagates variable value:
**Norinori**

**Face gadget**, for any open region:

**Corridor gadget**, propagates variable value:

- **Variable loop**
- **“false”**
Norinori

**Face gadget**, for any open region:

**Corridor gadget**, propagates variable value:

Variable loop

“false”

“true”
Norinori

**Face gadget**, for any open region:

**Corridor gadget**, propagates variable value:

Wires for both variable and its negation: connect to appropriate place of variable loop.
Bend gadget:
Bend gadget:

“false”
Bend gadget:

“false”

“true”
1-in-3 gadget:
1-in-3 gadget:

**At-most gadget** (connects corridors from two negated variables):
At-most gadget (connects corridors from two negated variables):
1-in-3 gadget:

At-most gadget (connects corridors from two negated variables):

both “true”
(both variables “false”)
Norinori

1-in-3 gadget:

At-most gadget (connects corridors from two negated variables):

- At most
- At most
- At most

both “true”
(both variables “false”)

different truth settings
Norinori

1-in-3 gadget:

At-most gadget (connects corridors from two negated variables):

- Both “true” (both variables “false”)
- Different truth settings
- Both “false” (both variables “true”)
1-in-3 gadget:

At-most gadget (connects corridors from two negated variables):

Clause gadget:
1-in-3 gadget:

At-most gadget (connects corridors from two negated variables):

Clause gadget:

no variable fulfills the clause
Norinori

1-in-3 gadget:

At-most gadget (connects corridors from two negated variables):

Clause gadget:

no variable fulfills the clause
1-in-3 gadget:

**At-most gadget** (connects corridors from two negated variables):

- Both “true” (both variables “false”)
- Different truth settings
- Both “false” (both variables “true”)

**Clause gadget:**

- No variable fulfills the clause
LITS
Place black squares in the polyominoes, such that the final board satisfies
Place black squares in the polyominoes, such that the final board satisfies
• The black squares form a connected polyomino.
Place black squares in the polyominoes, such that the final board satisfies
• The black squares form a connected polyomino.
• Each polyomino region contains a connected black tetromino.
Place black squares in the polyominoes, such that the final board satisfies
• The black squares form a connected polyomino.
• Each polyomino region contains a connected black tetromino.
• No two congruent tetrominoes are adjacent.
Place black squares in the polyominoes, such that the final board satisfies
• The black squares form a connected polyomino.
• Each polyomino region contains a connected black tetromino.
• No two congruent tetrominoes are adjacent.
• Black squares may not build 2x2 squares.
Place black squares in the polyominoes, such that the final board satisfies

- The black squares form a connected polyomino.
- Each polyomino region contains a connected black tetromino.
- No two congruent tetrominoes are adjacent.
- Black squares may not build 2x2 squares.
Place black squares in the polyominoes, such that the final board satisfies:

- The black squares form a connected polyomino.
- Each polyomino region contains a connected black tetromino.
- No two congruent tetrominoes are adjacent.
- Black squares may not build 2x2 squares.
Place black squares in the polyominoes, such that the final board satisfies
• The black squares form a connected polyomino.
• Each polyomino region contains a connected black tetromino.
• No two congruent tetrominoes are adjacent.
• Black squares may not build 2x2 squares.
Place black squares in the polyominoes, such that the final board satisfies
• The black squares form a connected polyomino.
• Each polyomino region contains a connected black tetromino.
• No two congruent tetrominoes are adjacent.
• Black squares may not build 2x2 squares.
Place black squares in the polyominoes, such that the final board satisfies

- The black squares form a connected polyomino.
- Each polyomino region contains a connected black tetromino.
- No two congruent tetrominoes are adjacent.
- Black squares may not build 2x2 squares.
LITS

Place black squares in the polyominoes, such that the final board satisfies:

- The black squares form a connected polyomino.
- Each polyomino region contains a connected black tetromino.
- No two congruent tetrominoes are adjacent.
- Black squares may not build 2x2 squares.

![Polyomino Example]

![Solution Example]
Place black squares in the polyominoes, such that the final board satisfies
• The black squares form a connected polyomino.
• Each polyomino region contains a connected black tetromino.
• No two congruent tetrominoes are adjacent.
• Black squares may not build 2x2 squares.
Place black squares in the polyominoes, such that the final board satisfies
• The black squares form a connected polyomino.
• Each polyomino region contains a connected black tetromino.
• No two congruent tetrominoes are adjacent.
• Black squares may not build 2x2 squares.
Place black squares in the polyominoes, such that the final board satisfies:

- The black squares form a connected polyomino.
- Each polyomino region contains a connected black tetromino.
- No two congruent tetrominoes are adjacent.
- Black squares may not build 2x2 squares.
Place black squares in the polyominoes, such that the final board satisfies

- The black squares form a connected polyomino.
- Each polyomino region contains a connected black tetromino.
- No two congruent tetrominoes are adjacent.
- Black squares may not build 2x2 squares.
Place black squares in the polyominoes, such that the final board satisfies
• The black squares form a connected polyomino.
• Each polyomino region contains a connected black tetromino.
• No two congruent tetrominoes are adjacent.
• Black squares may not build 2x2 squares.
Place black squares in the polyominoes, such that the final board satisfies

- The black squares form a connected polyomino.
- Each polyomino region contains a connected black tetromino.
- No two congruent tetrominoes are adjacent.
- Black squares may not build 2x2 squares.
**Theorem 2:**
Determining if a LITS board is solvable is NP-complete and counting the number of solutions is \#P-complete.
Theorem 2:
Determining if a LITS board is solvable is NP-complete and counting the number of solutions is #P-complete.

As for Norinori:
Theorem 2:
Determining if a LITS board is solvable is NP-complete and counting the number of solutions is \#P-complete.

As for Norinori:
Proof by reduction from PLANAR 1-IN-3-SAT.
Theorem 2: Determining if a LITS board is solvable is NP-complete and counting the number of solutions is #P-complete.

As for Norinori:
Proof by reduction from PLANAR 1-IN-3-SAT.
The properties of a final LITS board enforce unique feasible solutions for the following gadgets.
Theorem 2:
Determining if a LITS board is solvable is NP-complete and counting the number of solutions is \#P-complete.

As for Norinori:
Proof by reduction from PLANAR 1-IN-3-SAT.
The properties of a final LITS board enforce unique feasible solutions for the following gadgets.

Face gadget:
Theorem 2: Determining if a LITS board is solvable is NP-complete and counting the number of solutions is \#P-complete.

As for Norinori:
Proof by reduction from PLANAR 1-IN-3-SAT.
The properties of a final LITS board enforce unique feasible solutions for the following gadgets.

Face gadget:
Variable gadget:
Variable gadget:

Must be filled with a T.
Variable gadget:

Must be filled with a T.

“true”

“false”
Variable gadget: must be filled with a T.

Corridor gadget: linearly repeat this pattern.
NOT gadget:
NOT gadget:

Must be filled with an S.
LITS

NOT gadget:

Must be filled with an S.
The wires connected by the gadget always satisfy opposite truth assignments.
NOT gadget: The wires connected by the gadget always satisfy opposite truth assignments.

Must be filled with an S.

Bend gadget:
NOT gadget:

Must be filled with an S.

The wires connected by the gadget always satisfy opposite truth assignments.

Bend gadget:

Must be filled with a T.
The wires connected by the gadget always satisfy opposite truth assignments.

**NOT gadget:**

- Must be filled with an S.

**Bend gadget:**

- Must be filled with a T.
- “false”
- The other T wouldn’t connect to the incoming I.
NOT gadget:

Must be filled with an S.

The wires connected by the gadget always satisfy opposite truth assignments.

Bend gadget:

Must be filled with a T.

The other T wouldn’t connect to the incoming I.

Other I would leave S disconnected.
NOT gadget:

Must be filled with an S.

The wires connected by the gadget always satisfy opposite truth assignments.

Bend gadget:

Must be filled with a T.

The other T wouldn’t connect to the incoming I.

Other I would leave S disconnected.
LITS

NOT gadget:

Must be filled with an S.

The wires connected by the gadget always satisfy opposite truth assignments.

Bend gadget:

Must be filled with a T.

The other T wouldn't connect to the incoming I.

Other I would leave S disconnected.
The wires connected by the gadget always satisfy opposite truth assignments.

**NOT gadget:**
- Must be filled with an S.

**Bend gadget:**
- Must be filled with a T.
- "false"
- "true"
- The other T wouldn’t connect to the incoming I.

Other I would leave S disconnected.

Other I would result in 2x2 block.
Split gadget:
Split gadget:

Must be filled with an S or a T.
Split gadget:

Must be filled with an S or a T.

No position of T possible:
Split gadget:

Must be filled with an S or a T.

No position of T possible:
Split gadget:

Must be filled with an S or a T.

No position of T possible:
1-in-3 gadget:
1-in-3 gadget:

At-most gadget (Two C-shaped regions connect to variable corridors):
1-in-3 gadget:

*At-most gadget* (Two C-shaped regions connect to variable corridors):

Corridor of enforced I and T shapes that connect to an S or L of a corridor on that face.
1-in-3 gadget:

At-most gadget (Two C-shaped regions connect to variable corridors):

Corridor of enforced I and T shapes that connect to an S or L of a corridor on that face. Both variables truth setting that fulfils the clause.
1-in-3 gadget:

At-most gadget (Two C-shaped regions connect to variable corridors):

Corridor of enforced I and T shapes that connect to an S or L of a corridor on that face.

Both variables truth setting that fulfils the clause.

Both variables truth setting that does not fulfil the clause.
1-in-3 gadget:

**At-most gadget** (Two C-shaped regions connect to variable corridors):

- Both variables truth setting that fulfils the clause.
- Both variables truth setting that does not fulfil the clause.
- Only one variable fulfils the clause.

Corridor of enforced I and T shapes that connect to an S or L of a corridor on that face.
**1-in-3 gadget:**

**At-most gadget** (Two C-shaped regions connect to variable corridors):

- Both variables truth setting that fulfils the clause.
- Both variables truth setting that does not fulfil the clause.
- Only one variable fulfils the clause.

**Clause gadget:**
1-in-3 gadget:

At-most gadget (Two C-shaped regions connect to variable corridors):

Clause gadget:

All variables do not fulfil the clause
⇒ no tetromino in the **pink region** can be connected.
1-in-3 gadget:

At-most gadget (Two C-shaped regions connect to variable corridors):

Clause gadget:
Boards with Unique Solutions
Boards with Unique Solutions

$U_N(n,m) = \text{minimum number of regions among all } nxm \text{ Norinori boards with unique solutions}$
Boards with Unique Solutions

\[ U_N(n,m) = \text{minimum number of regions among all nxm Norinori boards with unique solutions} \]

\[ U_L(n,m) = \text{------"-------------"-------------"------------- LITS \hspace{50pt} ------"-------------"-------------} \]
Boards with Unique Solutions

$U_N(n,m) = \text{minimum number of regions among all nxm Norinori boards with unique solutions}$

$U_L(n,m) = \underbrace{\text{---}}_{\text{LITS}} \underbrace{\text{---}}_{\text{---}}$

(When undefined = 0)
Boards with Unique Solutions

$U_{\mathcal{N}}(n,m) =$ minimum number of regions among all $n \times m$ Norinori boards with unique solutions

$U_{\mathcal{L}}(n,m) =$  

(When undefined = 0)

**Theorem 3:** $U_{\mathcal{L}}(n,m) = 3$ for all $n \geq 10, m \geq 2$.

In other words, 3 regions suffice to completely determine an $n \times m$ LITS board, as long as $n \geq 10$ and $m \geq 2$. 
Boards with Unique Solutions

\[ U_{\mathcal{N}}(n,m) = \text{minimum number of regions among all nxm Norinori boards with unique solutions} \]

\[ U_{\mathcal{L}}(n,m) = \text{minimum number of regions among all nxm LITS boards with unique solutions} \]

(When undefined = 0)

**Theorem 3:** \( U_{\mathcal{L}}(n,m) = 3 \) for all \( n \geq 10, m \geq 2 \).

In other words, 3 regions suffice to completely determine an nxm LITS board, as long as \( n \geq 10 \) and \( m \geq 2 \).
 Boards with Unique Solutions

\[ U_n(n,m) = \text{minimum number of regions among all nxm Norinori boards with unique solutions} \]

\[ U_L(n,m) = \text{LITS} \]

(When undefined = 0)

**Theorem 3:** \( U_L(n,m) = 3 \) for all \( n \geq 10, m \geq 2 \).

In other words, 3 regions suffice to completely determine an nxm LITS board, as long as \( n \geq 10 \) and \( m \geq 2 \).
Boards with Unique Solutions

\[ U(n,m) = \text{minimum number of regions among all nxm Norinori boards with unique solutions} \]

\[ U_L(n,m) = \frac{n}{3} \quad \text{LITS} \quad \frac{n}{4} \quad \text{for all n} \geq 10, m \geq 2. \]

In other words, 3 regions suffice to completely determine an nxm LITS board, as long as \( n \geq 10 \) and \( m \geq 2 \).

\textbf{Theorem 3:} \( U_L(n,m) = 3 \) for all \( n \geq 10, m \geq 2 \).

\textbf{Theorem 4:}

1. \( U_N(n, 1) = 0 \) for \( n \neq 2 \mod 3 \)

2. \( U_N(n, 1) = \frac{n+1}{3} \) for \( n \equiv 2 \mod 3 \)

3. \( U_N(n, 2) \leq \left\lfloor \frac{n}{4} \right\rfloor \) for \( n \geq 3 \)

4. \( U_N(n, m) = 3 \) for all \( n \geq 5, m \geq 3 \).
Boards with Unique Solutions

\(U_N(n,m)\) = minimum number of regions among all \(n \times m\) Norinori boards with unique solutions

\(U_L(n,m) = \frac{n+1}{3}\) for \(n \equiv 2 \mod 3\)

Theorem 3: \(U_L(n,m) = 3\) for all \(n \geq 10, m \geq 2\).
In other words, 3 regions suffice to completely determine an \(n \times m\) LITS board, as long as \(n \geq 10\) and \(m \geq 2\).

Theorem 4:

1. \(U_N(n,1) = 0\) for \(n \neq 2 \mod 3\)
2. \(U_N(n,1) = \frac{n+1}{3}\) for \(n \equiv 2 \mod 3\)
3. \(U_N(n,2) \leq \left\lceil \frac{n}{4} \right\rceil\) for \(n \geq 3\)
4. \(U_N(n,m) = 3\) for all \(n \geq 5, m \geq 3\).
Boards with Unique Solutions

\[ U_{N}(n,m) = \text{minimum number of regions among all nxm Norinori boards with unique solutions} \]

\[ U_{L}(n,m) = \begin{cases} 
1 & \text{when undefined} \\
0 & \text{otherwise}
\end{cases} \]

(When undefined = 0)

**Theorem 3:** \( U_{L}(n,m) = 3 \) for all \( n \geq 10, m \geq 2 \).

In other words, 3 regions suffice to completely determine an nxm LITS board, as long as \( n \geq 10 \) and \( m \geq 2 \).

**Theorem 4:**

1. \( U_{N}(n, 1) = 0 \) for \( n \not\equiv 2 \mod 3 \)
2. \( U_{N}(n, 1) = \frac{n+1}{3} \) for \( n \equiv 2 \mod 3 \)
3. \( U_{N}(n, 2) \leq \left\lceil \frac{n}{4} \right\rceil \) for \( n \geq 3 \)
4. \( U_{N}(n, m) = 3 \) for all \( n \geq 5, m \geq 3 \).
Boards with Unique Solutions

\( U(n,m) = \text{minimum number of regions among all } nxm \text{ Norinori boards with unique solutions} \)

\( U_L(n,m) = \) ———“———“———“——— LITS ———“———“———

(When undefined =0)

**Theorem 3:** \( U_L(n,m) = 3 \) for all \( n \geq 10, m \geq 2 \).

In other words, 3 regions suffice to completely determine an \( nxm \) LITS board, as long as \( n \geq 10 \) and \( m \geq 2 \).

**Theorem 4:**

1. \( U_N(n, 1) = 0 \) for \( n \neq 2 \mod 3 \)
2. \( U_N(n, 1) = \frac{n+1}{3} \) for \( n \equiv 2 \mod 3 \)
3. \( U_N(n, 2) \leq \left\lceil \frac{n}{4} \right\rceil \) for \( n \geq 3 \)
4. \( U_N(n, m) = 3 \) for all \( n \geq 5, m \geq 3 \).
THANK YOU.

★ Determining if a Norinori board is solvable is NP-complete and counting the number of solutions is \#P-complete.
★ Determining if a LITS board is solvable is NP-complete and counting the number of solutions is \#P-complete.
★ Bounds on the minimum number of regions among all nxm Norinori/LITS boards with unique solutions.