# *patoLogic* solves

# STARBATTLE QUACKLACTICA

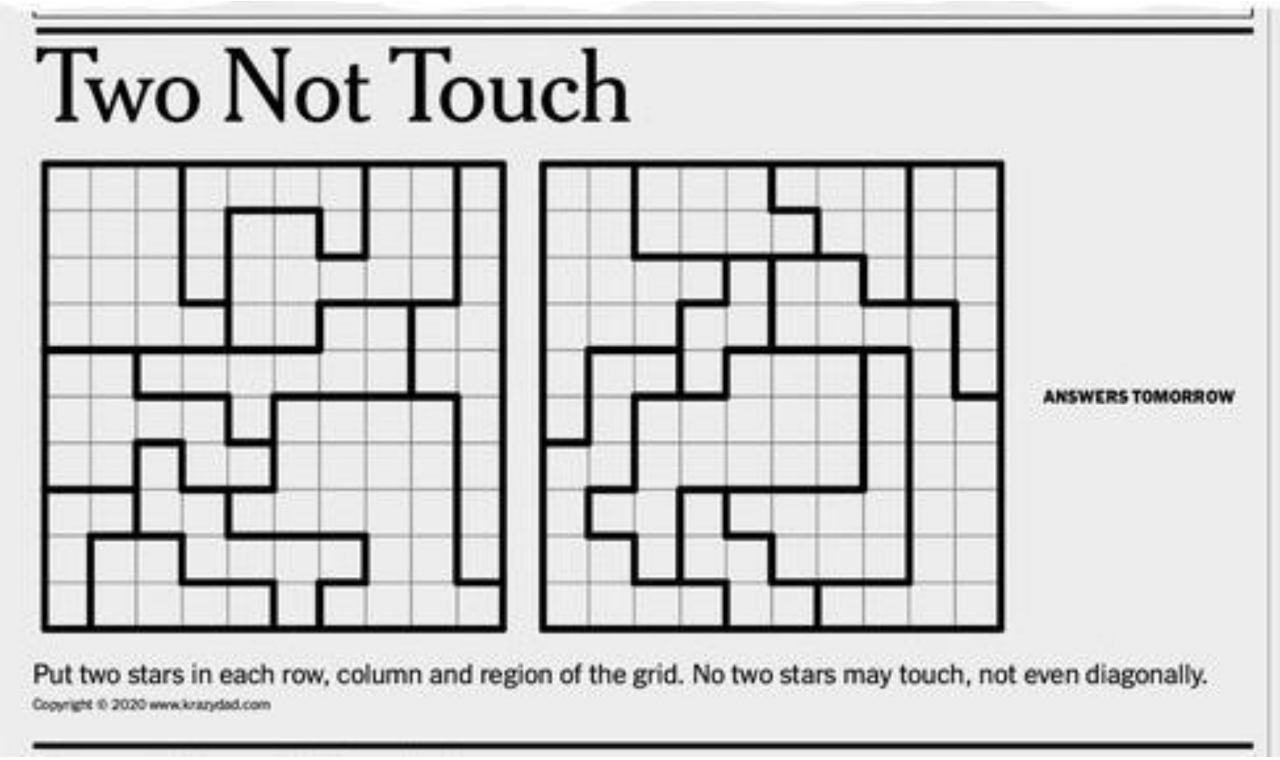
## Creating and Solving Battle Star Games With Quanutm Annealig

Diogo Cruz, Duarte Magano, Óscar Amaro, Sagar Pratapsi

MIT IQuHACK 2021

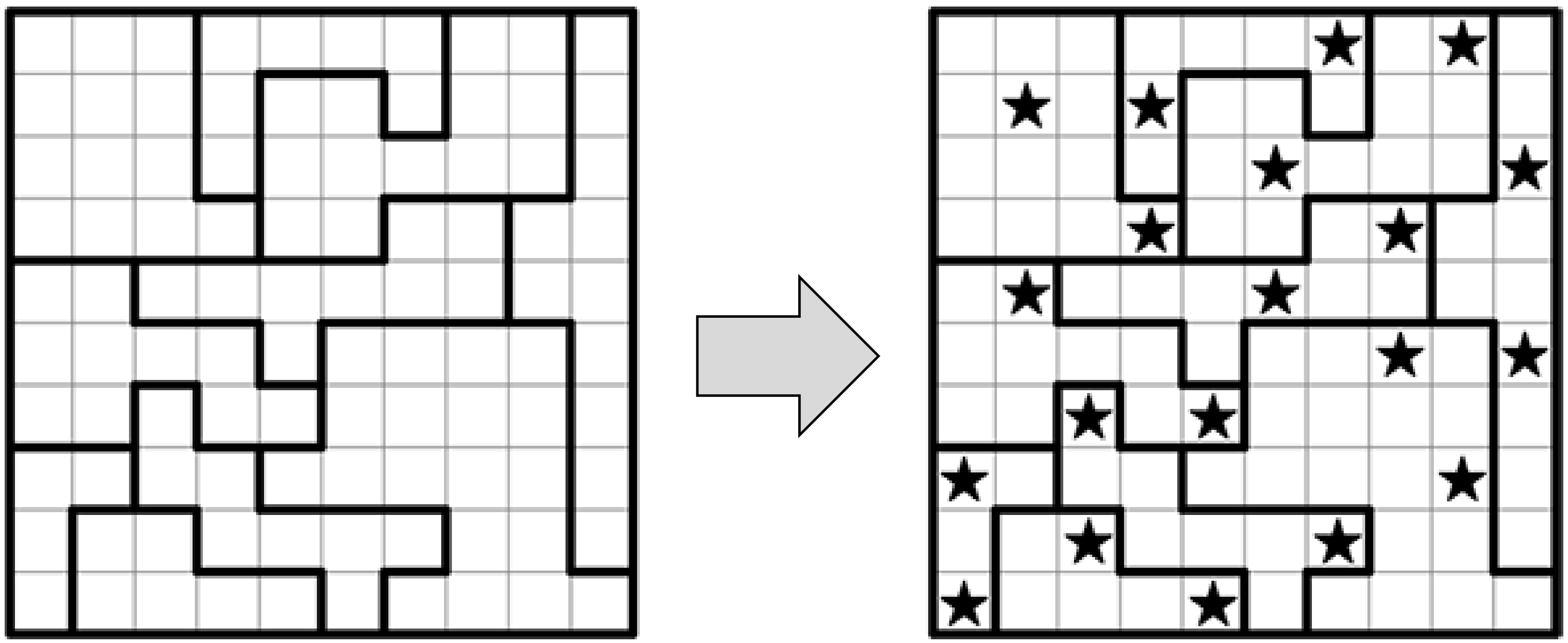
# An Addictive Puzzle

Two not Touch, or Star Battle, is a game regularly published in the puzzle section of the New York Times.



# Example

This one is considered an easy one:



# Our Challenge

Given an $N \times N$ grid divided into $N$ regions:

```
grid = np.array([[0,1,1,1,1],
                 [0,1,1,1,2],
                 [0,0,0,2,2],
                 [3,0,3,2,2],
                 [3,3,3,4,4]])
```

, find a distribution of stars that satisfies the constraints of the Two Not Touch:

```
solution = np.array([[0,1,0,0,0],
                     [0,0,0,0,1],
                     [0,0,1,0,0],
                     [1,0,0,0,0],
                     [0,0,0,1,0]])
```

# Create Puzzle

Assume that we are given a puzzle which is solvable:

*Import generate_problem*

*Grid = function(n)*

We generate such a input with a  classical algorithm (for now!).

Challenge:

Solve $S$ Not Touch
with Quantum Annealing!

# Solution Constraints

Our solution needs to satisfy the following constraint:

- Each row contains exactly $S$ stars
- Each column contains exactly $S$ stars
- Each delimited region contains exactly $S$ stars

We call these *region constraints*

- All stars must be neighbour-free

# QUBO Variables

Our variables are

$$x_{ij} \in \{0,1\}$$

$$i, j \in \{1, \dots, N\}$$

The variable $x_{ij}$ is $1$ if the cell $(i, j)$ has a star and $0$ otherwise.

# Encoding Region Constraints

Let $R = \{(i_1, j_1), (i_2, j_2) \dots\}$ be a region (line, column, or other delimited region).

We impose that $R$ only has $S$ stars by including the term

$$H \mathrel{+}= \left(S - \sum_{(i,j)\in R} x_{ij}\right)^2$$

# Encoding Neighbour Constraints

We impose that a star on cell $(i, j)$ does not have a neighbour star by including term

$$H \mathrel{+}= x_{ij} \left( \sum_{(i',j')\ neighbour\ to\ (i,j)} x_{i'j'} \right)$$

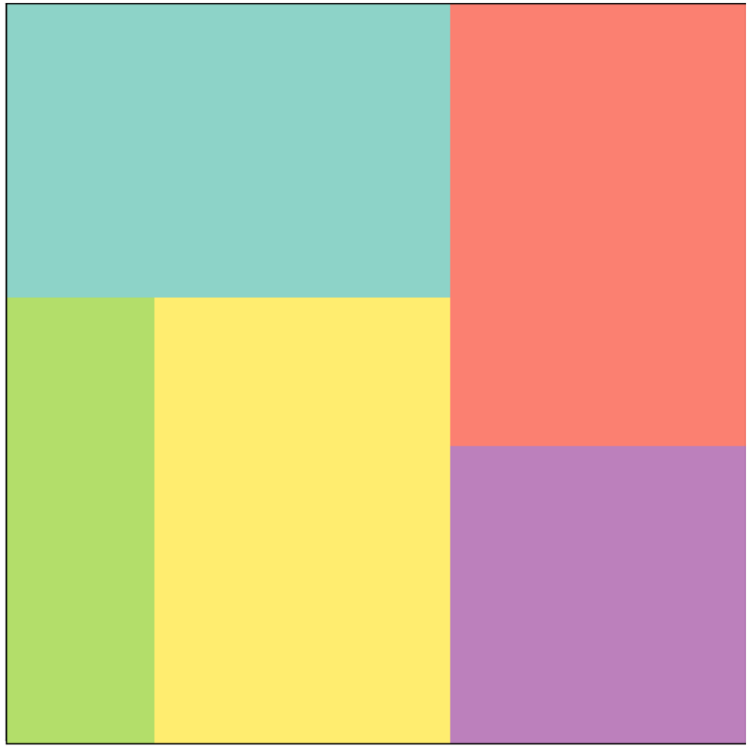
# QUBO Hamiltonian

We use

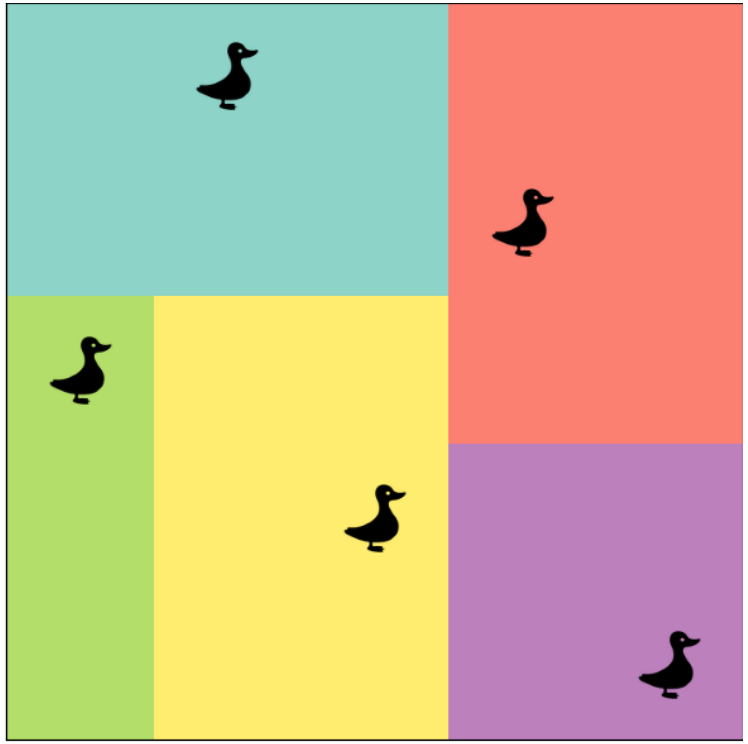
$$H = \sum_{R} \left( S - \sum_{(i,j) \in R} x_{ij} \right)^2 + \sum_{(i,j)} x_{ij} \left( \sum_{(i',j')\ neighbour\ to\ (i,j)} x_{i'j'} \right)$$

# Results: $N = 5$

Input:

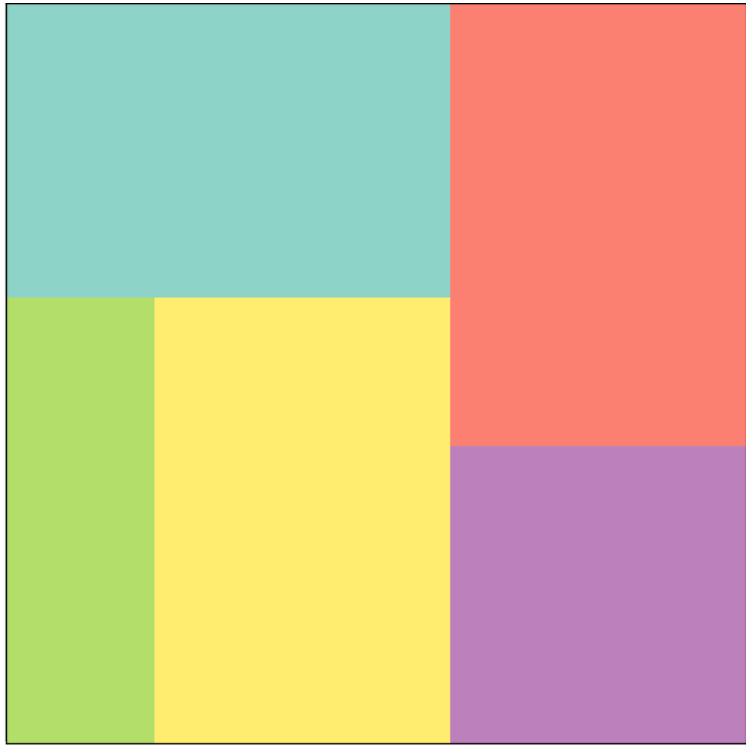


Ouput:

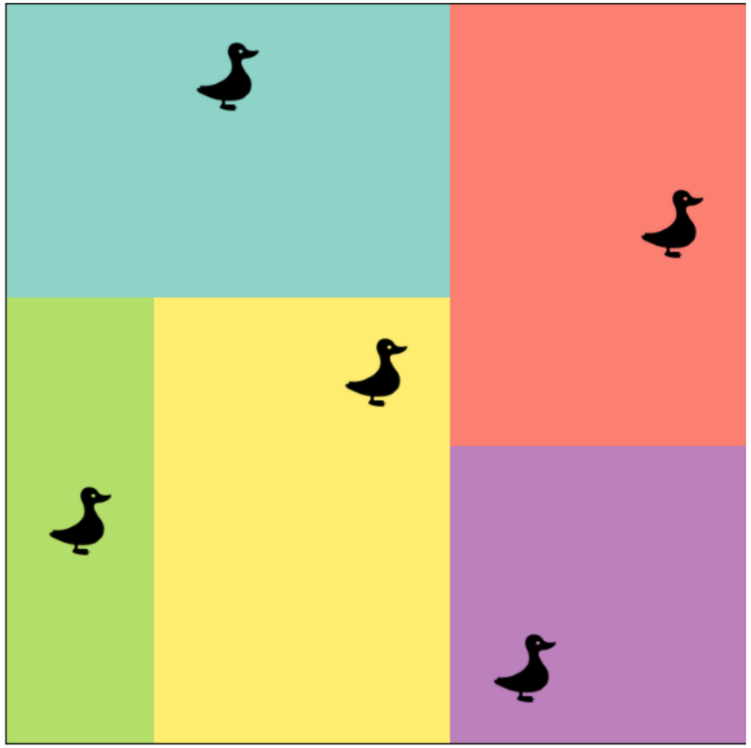


# Results: $N = 5$

Input:

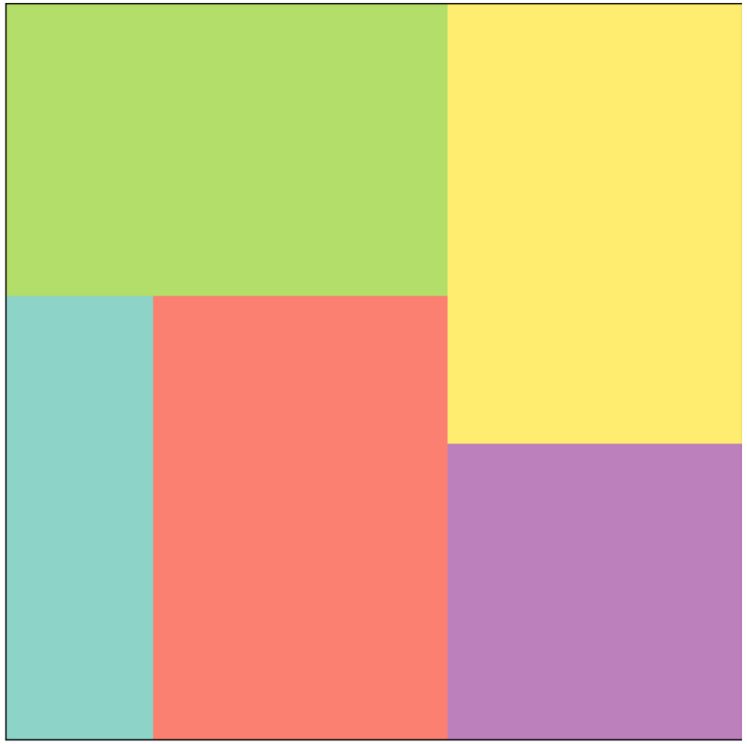


Ouput:

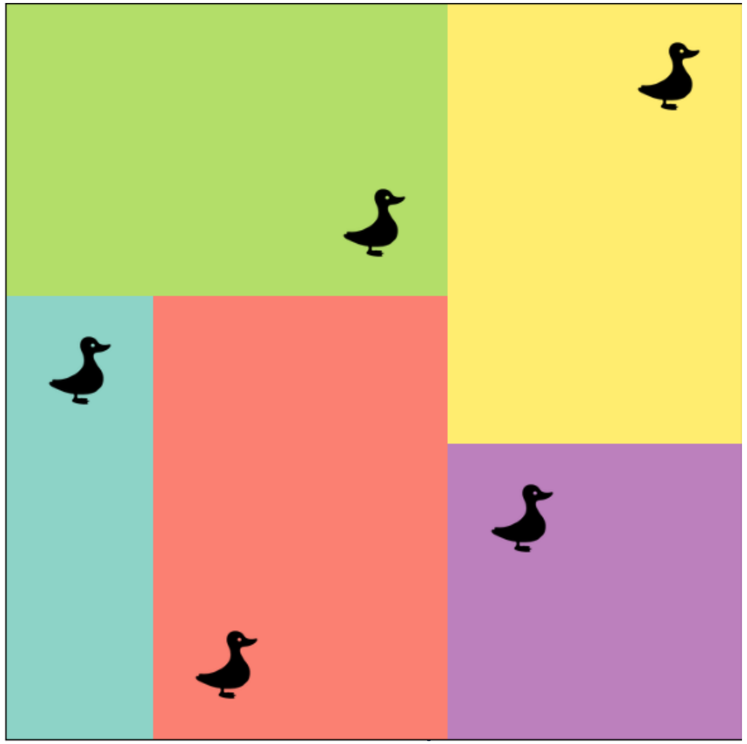


# Results: $N = 5$

Input:

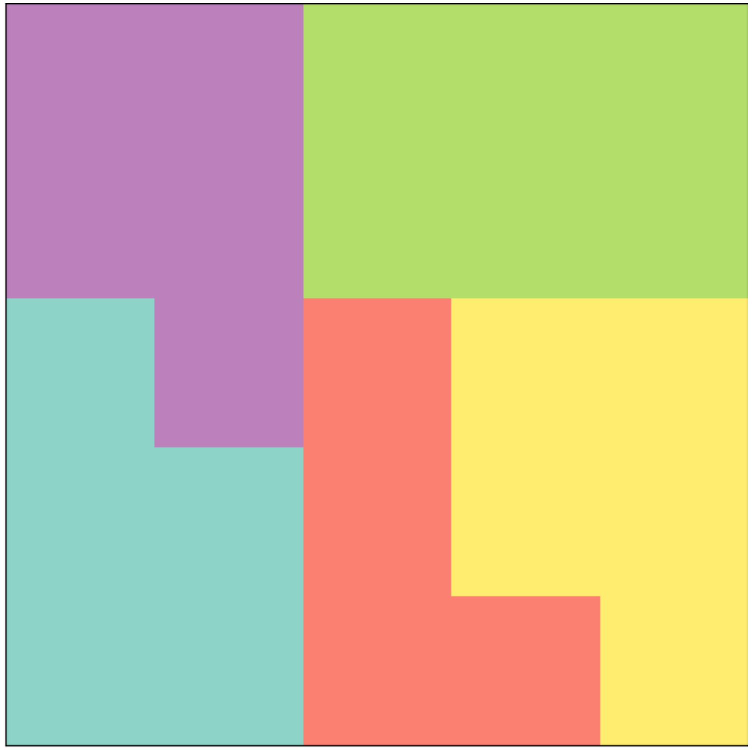


Ouput:

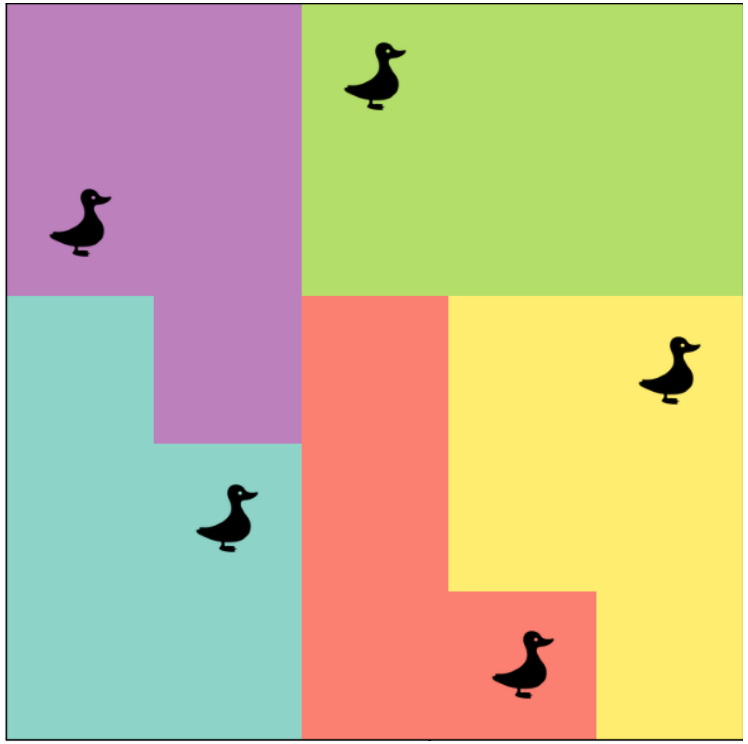


# Results: $N = 5$

Input:

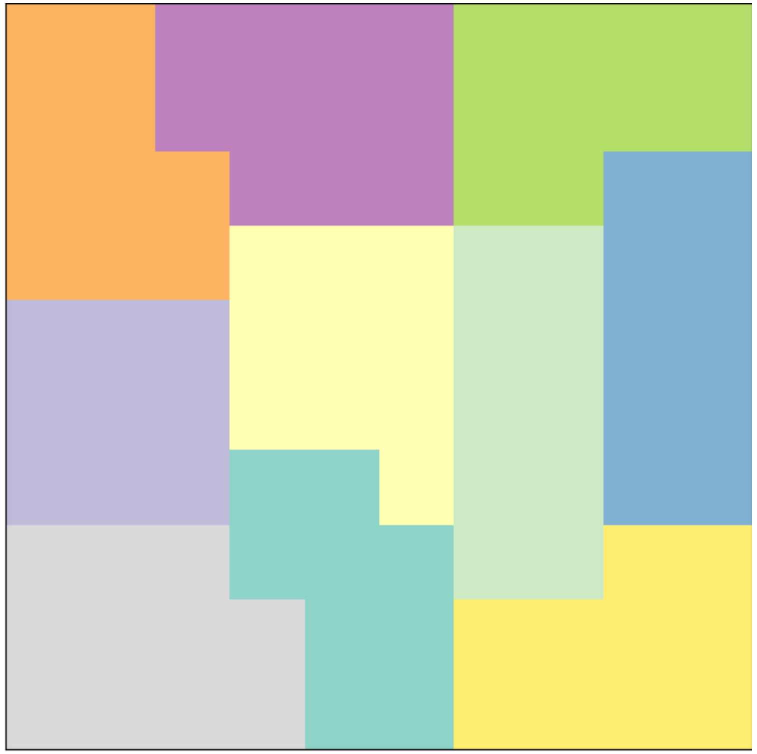


Ouput:

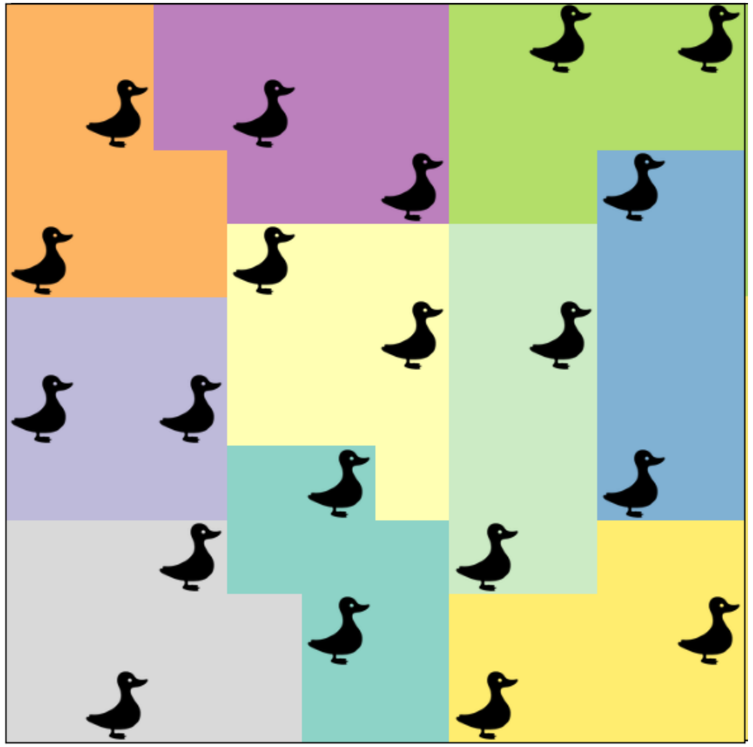


# Results: $N = 10$

Input:

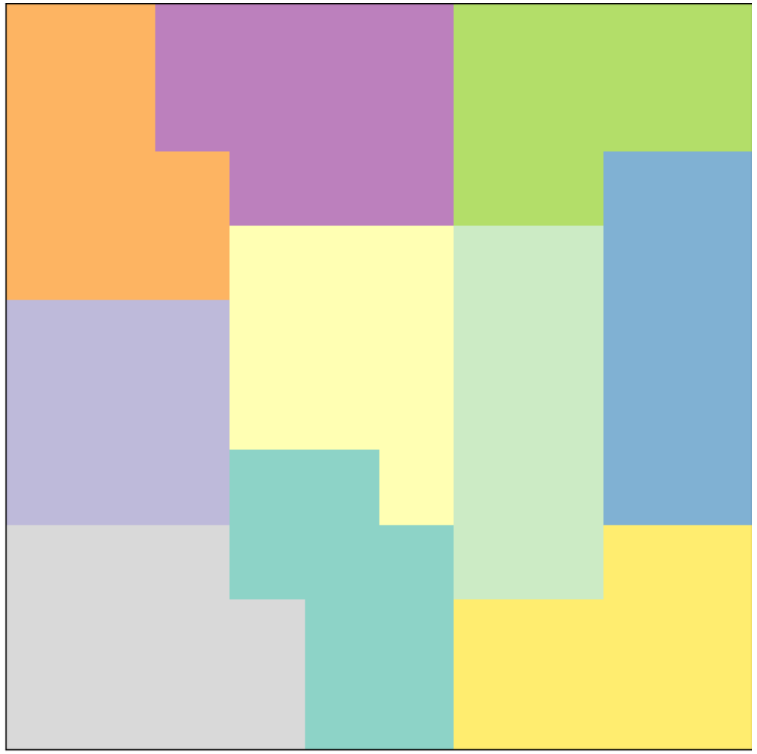


Ouput:

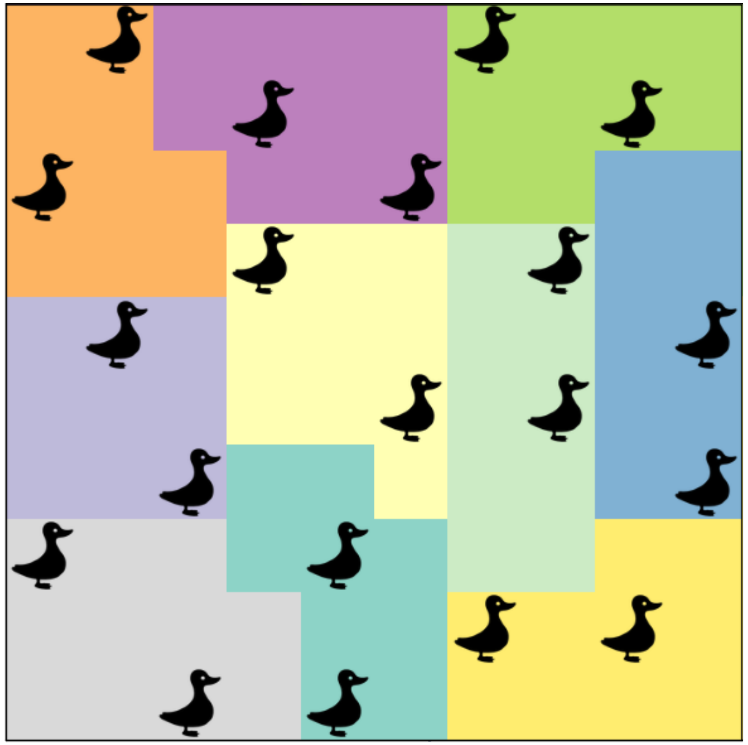


# Results: $N = 10$

Input:

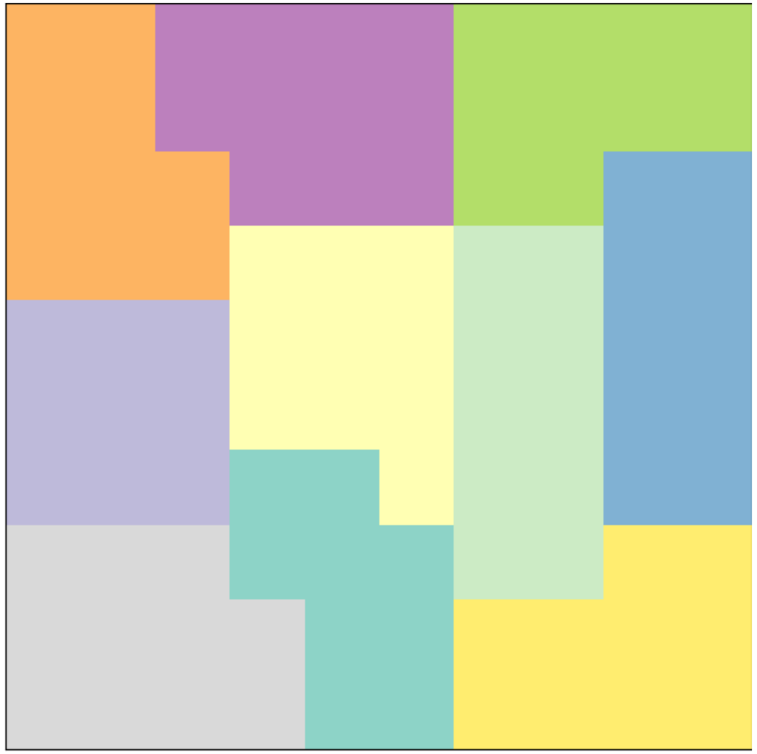


Ouput:

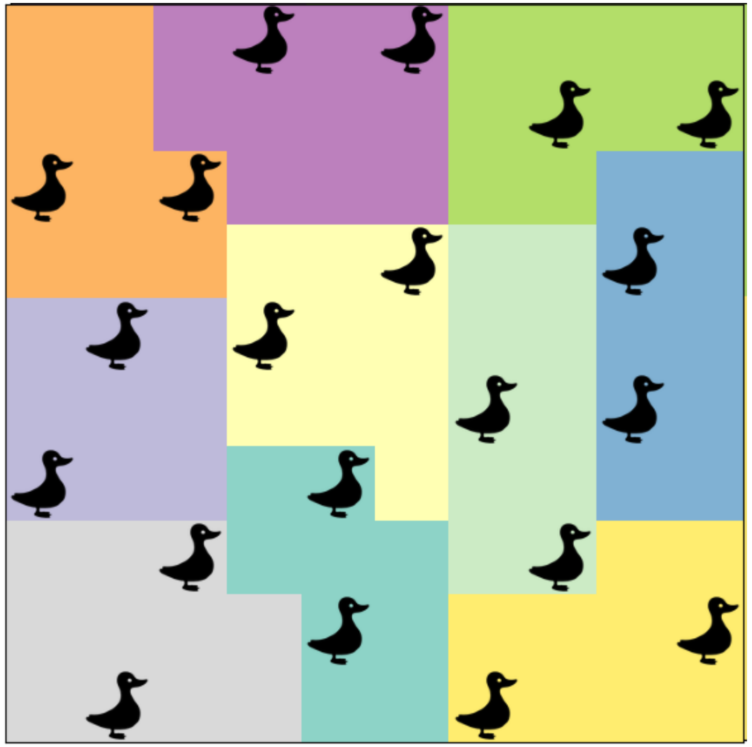


# Results: $N = 10$

Input:



Ouput:



# Reverse Challenge:

# Generate Solvable Puzzle with Quantum Annealing

# Annealing in Two Parts

Given $N$,

1. Generate distribution of stars such that
   - Each row and each column contains exactly $S$ stars.
   - All stars are neighbour free
2. Generate $N$ regions  such that
   - Each region contains exactly $S$ stars
   - Each regions is connected

These regions will form a valid puzzle.

# QUBO for Star Distribution

This is the same Hamiltonian as to solve the problem, but without the delimited regions contraints

$$H = \sum_i \left( S - \sum_j x_{ij} \right)^2 + \sum_j \left( S - \sum_i x_{ij} \right)^2$$

$$+ \sum_{(i,j)} x_{ij} \left( \sum_{(i',j')\ neighbour\ to\ (i,j)} x_{i'j'} \right)$$

Region generation is a more difficult problem!

# QUBO Variables

Our variables are

$$x_{ij}^{c} \in \{0,1\}$$

$$i, j, c \in \{1, \dots, N\}$$

The variable $x_{ij}^{c}$ is $1$ if the cell $(i, j)$ belongs to region $c$ and $0$ otherwise.

# Encoding Region Generation Constraints

Each cell $(i, j)$ belongs to a unique region

$$H \mathrel{+}= \left(1 - \sum_{c} x_{ij}^{c}\right)^2$$

# Encoding Region Generation Constraints

Each region $c$ has $S$ stars

$$H \mathrel{+}= \left(S - \sum_{(i,j)\in stars} x_{ij}^{c}\right)^{2}$$

# Encoding Region Generation Constraints

How to enforce connectiveness?

We can favour neighbour cells of the same colour

$$H \mathrel{+}= \sum_{(i,j)} \left(x_{ij}^{c}\left(x_{i+1j}^{c} + x_{ij+1}^{c} + x_{i-1j}^{c} + x_{ij-1}^{c}\right) - 2.5\right)^{2}$$

This does not provide guarantees – it is compatible with very small regions…

# Encoding Region Generation Constraints

So, we add the additional condition that each region $c$ should have $N$ cells

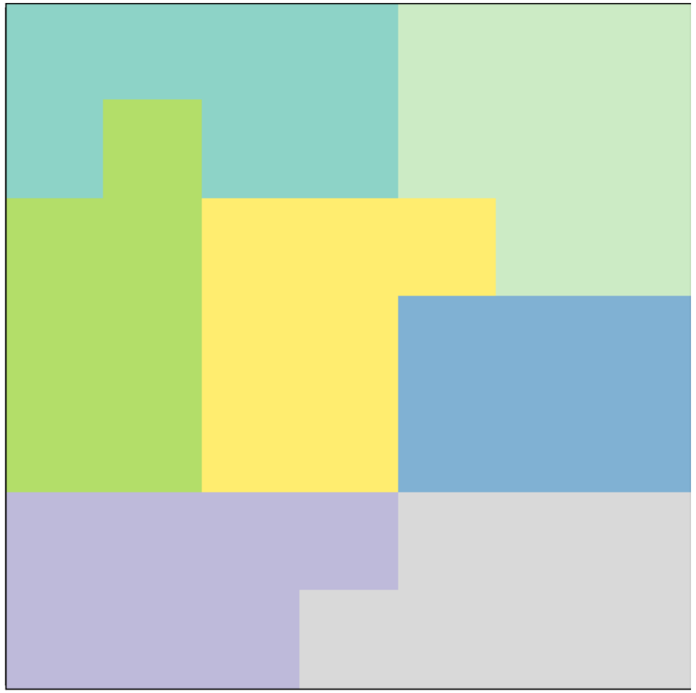
$$H \mathrel{+}= \left( N - \sum_{(i,j)} x_{ij}^{c} \right)^2$$

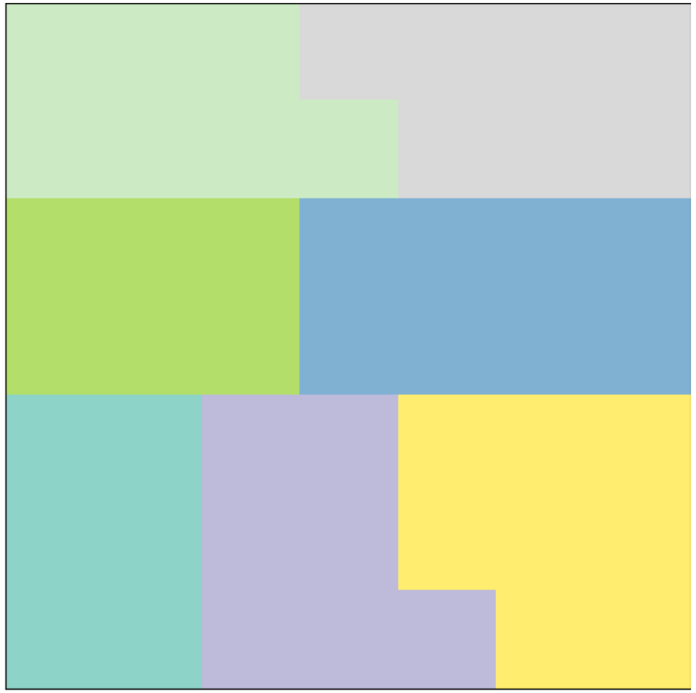
# QUBO for Region Generation

We use the Hamiltonian

$$H = \sum_{(i,j)} \left(1 - \sum_{c} x_{ij}^{c}\right)^2 + \sum_{c} \left(S - \sum_{(i,j)\in stars} x_{ij}^{c}\right)^2$$
$$+ \sum_{(i,j)} \left(x_{ij}^{c}\left(x_{i+1j}^{c} + x_{ij+1}^{c} + x_{i-1j}^{c} + x_{ij-1}^{c}\right) - 2.5\right)^2 + \sum_{c} \left(N - \sum_{(i,j)} x_{ij}^{c}\right)^2$$

# Puzzles with $N = 12$





# Puzzles with $N = 12$