Sliding Blocks Puzzle Solver
Technical Notes

1 Problem Description

The sliding blocks puzzle consists of numbered tiles on an $N \times N$ grid with one empty space. Tiles adjacent to the empty space can slide into it. The goal is to arrange tiles in ascending order with the empty space in the bottom-right corner.

1.1 Solvability

A configuration is solvable if and only if:

- For N odd: number of inversions is even
- For N even: (inversions + empty row from bottom) is odd

2 PLNE Model (Programme Linéaire en Nombres Entiers)

Pure integer linear programming formulation.

2.1 Decision Variables

- $x_{i,j,t} \in \{0,1\}$: tile i is at position j at time t
- $m_{i,j,t} \in \{0,1\}$: tile i moves from position j at time t

where $i, j \in \{0, \ldots, N^2 - 1\}$ and $t \in \{0, \ldots, T\}$

2.2 Constraints

Position Uniqueness:

$$\sum_j x_{i,j,t} = 1 \quad \forall i, t \quad \text{(each tile at exactly one position)} \tag{1}$$

$$\sum_i x_{i,j,t} = 1 \quad \forall j, t \quad \text{(each position has exactly one tile)} \tag{2}$$

Initial and Goal States:

$$x_{i,\text{pos}_{\text{init}}(i),0} = 1 \quad \forall i \tag{3}$$

$$x_{i,\text{pos}_{\text{goal}}(i),T} = 1 \quad \forall i \tag{4}$$

Move Constraints:

$$m_{i,j,t} \leq x_{i,j,t} \quad \forall i, j, t \tag{5}$$

$$\sum_{i,j} m_{i,j,t} = 1 \quad \forall t \tag{6}$$

$$x_{i,j,t+1} \leq x_{i,j,t} + \sum_{k \in \text{neighbors}(j)} m_{i,k,t} \quad \forall i \neq 0, j, t \tag{7}$$

$$x_{i,j,t} - x_{i,j,t+1} \leq x_{0,j,t} + m_{i,j,t} \quad \forall i \neq 0, j, t \tag{8}$$

2.3 Objective

$$\text{minimize: } \sum_{t,i,j} m_{i,j,t} \tag{9}$$

3 PLM Model (Programme Linéaire Mixte)

Mixed integer-continuous formulation for improved performance.

3.1 Additional Variables

- $\text{dist}_{i,t} \in \mathbb{R}^+$: Manhattan distance of tile i from goal at time t

3.2 Additional Constraints

Manhattan Distance:

$$\text{dist}_{i,t} \geq \sum_{j} (|\text{row}(j) - \text{row}_{\text{goal}}(i)| + |\text{col}(j) - \text{col}_{\text{goal}}(i)|) \times x_{i,j,t} \quad \forall i \neq 0, t \tag{10}$$

3.3 Objective

$$\text{minimize: } \sum_{t,i,j} m_{i,j,t} + 0.01 \times \sum_{i} \text{dist}_{i,T} \tag{11}$$

The distance penalty guides LP relaxation toward better solutions.

4 Model Comparison

Aspect	**PLNE**	**PLM**
Variables	All binary	Binary + continuous
Performance (3×3)	0.2–1.0s	0.1–0.8s
Performance (4×4)	5–60s	2–30s

Table 1: Comparison of PLNE and PLM models

Note: I was unable to get the 4×4 puzzle to work since the constraint count is higher than what my license allows.

5 Implementation Architecture

5.1 Core Classes

PuzzleState: Board representation, move generation, solvability checking

GurobiSolver: Optimization solver with mode selection (PLNE/PLM)

- Iterative deepening: starts with $T = 1$, increments until solution found (max $T = 12$)
- Guarantees optimal solution within horizon limit
- *Note:* Size-limited licenses may hit limits at $T \approx 9$–12 depending on puzzle complexity

PuzzleWidget: PyQt6 visualization with animations

MainWindow: UI with solver mode selector, controls, statistics display

6 Usage

```
pip install -r requirements.txt
python main.py
```

Select puzzle size (3×3 or 4×4), choose solver mode (PLNE or PLM), shuffle, and solve.

7 Testing

```
python -m pytest test_puzzle_state.py -v
python -m pytest test_gurobi_solver.py -v
python verify_solver.py
```