# Aritificial Intelligence CSCI611
# Project I Report

Fatulla Bashirov, Elmar Mammadov

## Description

The project aims to solve the famous N-puzzle problem which consists of (nxn)-matrix which is filled with elements $1 \dots N$ and one element denoted by 0 representing empty cell. Problem formulation:

- **States**: represents the position of all tiles in a matrix
- **Initial state**: any state
- **Goal state**: $0 \dots n \times n$
- **Actions**: movement of the blank space Left, Right, Up, Down, depending on the position of the empty state
- **Goal test**: Checks the current matrix against the matrix of the goal state
- **Path cost**: cost of each step is 1

Thus, given an input, the problem is to move tiles according to actions defined so as to get the configuration represented in the goal state. Current project uses $A*$ algorithm to solve the problem. Several heuristics are implemented - Manhattan Distance, Manhattan Distance + Linear Conflict distance, Mismatched tiles, Gasching's distance - for comparison. The most effective heuristic is Manhattan Distance + Linear Conflict. Simply put, Linear Conflict occurs when two tiles are in the same goal row or column but in reversed order, so they block each other and need extra moves. In the implementation, the function computes the linear conflict heuristic by counting pairs of tiles in the same row or column that are in their goal line but in reversed order, and returns twice that count to estimate the minimum extra moves needed beyond Manhattan distance. This is due to the fact that this heuristic gives larger result that the other listed and since all admissible heuristics never overestimate, the Manhattan Distance + Linear Conflict is closer to the true cost of solving the problem. Proof that this huristics is consistent and hence admissible can be found in [1]. The constraints of the problem are $3 <= n <= 5$. The code can be accessed in the following repository `https://github.com/elmarMamedov485/AI_project_I/tree/main`.

Example matrices:

| Size | Initial state | Goal state |
|---|---|---|
| 3x3 | $\begin{bmatrix} 1 & 0 & 2 \\ 3 & 5 & 4 \\ 6 & 8 & 7 \end{bmatrix}$ | $\begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{bmatrix}$ |
| 4x4 | $\begin{bmatrix} 5 & 1 & 15 & 7 \\ 8 & 4 & 2 & 11 \\ 0 & 3 & 6 & 14 \\ 12 & 9 & 10 & 13 \end{bmatrix}$ | $\begin{bmatrix} 0 & 1 & 2 & 3 \\ 4 & 5 & 6 & 7 \\ 8 & 9 & 10 & 11 \\ 12 & 13 & 14 & 15 \end{bmatrix}$ |
| 5x5 | $\begin{bmatrix} 0 & 1 & 2 & 3 & 4 \\ 5 & 6 & 16 & 8 & 9 \\ 10 & 11 & 12 & 13 & 14 \\ 15 & 7 & 17 & 19 & 18 \\ 20 & 21 & 22 & 23 & 24 \end{bmatrix}$ | $\begin{bmatrix} 0 & 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 & 9 \\ 10 & 11 & 12 & 13 & 14 \\ 15 & 16 & 17 & 18 & 19 \\ 20 & 21 & 22 & 23 & 24 \end{bmatrix}$ |

Comparison of the heuristics:

| Test | Manhattan + Linear Conflict (nodes / time s) | Manhattan (nodes / time s) | Gasching (nodes / time s) | Misplaced (nodes / time s) |
|---|---|---|---|---|
| 3x3 | 501 / 0.004778 | 703 / 0.005620 | 2628 / 0.086516 | 1968 / 0.015341 |
| 4x4 | 35530 / 0.504591 | 100773 / 0.816537 | - | - |
| 5x5 | 404253 / 8.389636 | 1091149 / 15.692326 | - | - |

We can see that simple heuristics such as Gasching and Misplaced distances are only effective for 8-puzzle while, for 15 and 25 puzzles their time exceeded 40s. Meanwhile, Manhattan distance provides results for all three instances, but uses more memory and is slower. Finally, Manhattan + Linear Conflict heuristics is the most effective, as it is closer to the true cost than other heuristics. However, this heuristic struggles with random $4 \times 4$ and $5 \times 5$ matrices as search tree becomes too large and the algorithm may process up to 8415013 nodes in 158.042681 seconds. All heuristics fail to compute 35-puzzle or bigger.

# References

[1] Othar Hansson, Andrew E. Mayer, and Mordechai M. Yung. *Generating Admissible Heuristics by Criticizing Solutions to Relaxed Models*. Tech. rep. CUCS-219-85. Department of Computer Science, Columbia University, 1985.