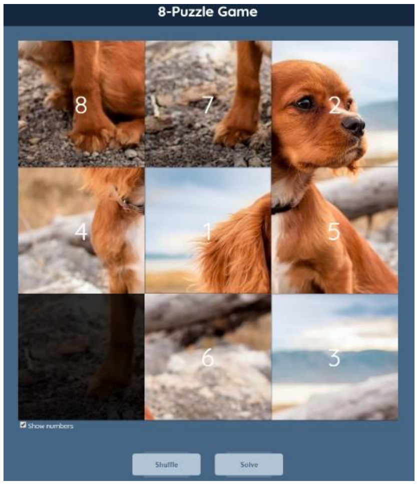
Name: Hoang Do

Partners: William Awad

Student ID: 01521888

N-Puzzle with AI

In this project, we will try to create the N-puzzle with AI that we learned from class where n can be 8, 15, 24 or 35. We decided to choose this topic since we saw a lot of examples about 8-puzzle in class and we know how it works. In this project, we not only use the numbers moving around but also include the picture or the user can choose their own picture as they want to create the game. Example:



(https://murhafsousli.github.io/8puzzle/)

The puzzle game has the initial state and the goal state as the picture below. The state space is depending on the number of n: n^2!. For example: if n = 3 (8-puzzle), the state space is 9! = 362880; if n = 4 (15-puzzle), the state space is 16! = 20922789888000. But only have of these states are reachable from any given states [Johnson and Storey, 1879]. Hence if n =3, there is 9!/2 solvable problems instance.

6	🟥	1
7	2	5
3	8	4

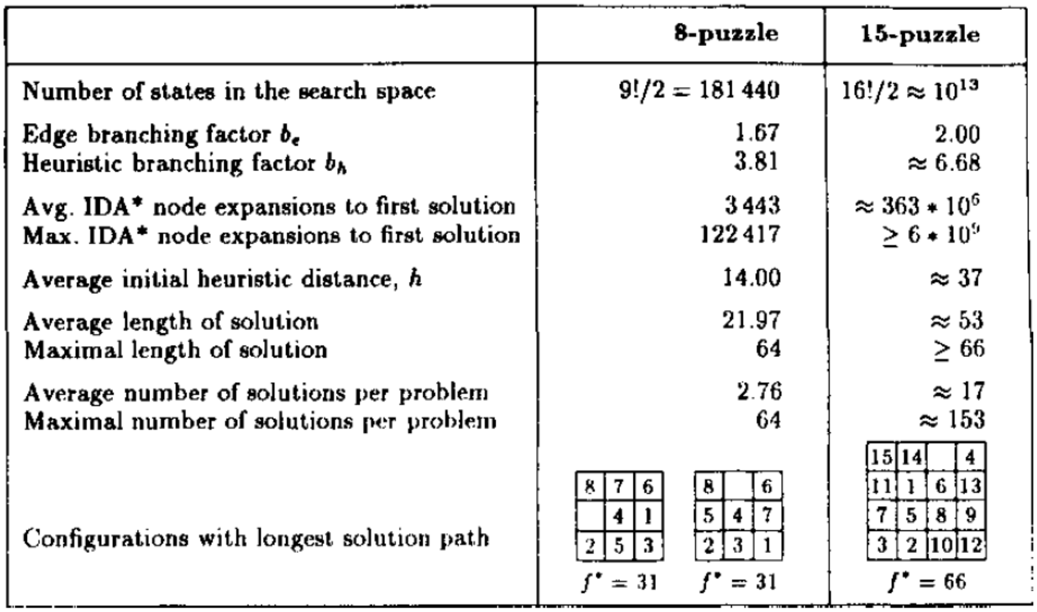
Init State

🟥	1	2
3	4	5
6	7	8

Goal State

🟥	1	2
3	4	5
6	8	7

Unreachable State



	8-puzzle		15-puzzle
Number of states in the search space	$9!/2 = 181\,440$		$16!/2 \approx 10^{13}$
Edge branching factor b_e	1.67		2.00
Heuristic branching factor b_h	3.81		≈ 6.68
Avg. IDA* node expansions to first solution	3 443		$\approx 363 * 10^6$
Max. IDA* node expansions to first solution	122 417		$\geq 6 * 10^9$
Average initial heuristic distance, h	14.00		≈ 37
Average length of solution	21.97		≈ 53
Maximal length of solution	64		≥ 66
Average number of solutions per problem	2.76		≈ 17
Maximal number of solutions per problem	64		≈ 153
Configurations with longest solution path	8 7 6 □ 4 1 2 5 3 $f^* = 31$	8 □ 6 5 4 7 2 3 1 $f^* = 31$	15 14 □ 4 11 1 6 13 7 5 8 9 3 2 10 12 $f^* = 66$

Table 1: Summary: The 8-puzzle results are exact; the 15-puzzle data is based on Korf's random problem set

In order to move from one state to another state or from initial state to goal state, we swap the empty tile (red color) with other tiles number (blue color) from the initial state to achieve the goal state. Those moving state are deterministic transitions and each move cost a unit of time (second).

Task environment	Observable	Deterministic	Episodic	Static	Discrete	Agent
N-Puzzle	Fully	Deterministic	Episodic	Static	Discrete	Single

N-Puzzle is fully observable because the agent can access to all information in the environment relevant to its tasks. It is also a deterministic environment because the outcome can be determined base on the current state. It is episodic because there is only one goal for this problem and we will try to reach that goal. The environment is not changing and always observable making it static. It is a discrete single agent because the environment is a fixed location with only one changing variable(empty space).

We will only be working in A* and IDA* as they are the most relevant to our task. Along with this we will be working on some heuristic as well as the node evaluation and the running time in order to have a high performance rate.

By accomplishing our goal we hope to have a better understanding about the process of algorithm development. We will represent our project through an interactive game that accurately demonstrates our understanding of AI.

The outcome of this project is a game that uses a simple user interface that is gui based. Within this interface you will have the option to solve the puzzle yourself at different levels as well as having the option to have our AI solve it for you. As a stretch goal we hope to allow the user to use their own photo in the game.

When the AI solves the puzzle you will be given a breakdown of the overall performance based on what was discussed in class on how to determine the most effective and efficient state traversal from initial state to goal state.

We hope to have a good pseudo code and interface for the over all project within two weeks. This will be done by dividing the initial design period among the members in the team. From there we will divide the work based on the comfortability of the different members. The schedule will be as follows:

Week1-2:

Pseudo-Code and User Interface Development

Week3:

Code development for initial program base

Week4:

Debug and expand into GUI

Week5:

Final Debug and Presentation completion

Reference

1. Bhasin, Harsh, and Neha Singia. “Genetic Based Algorithm for N –Puzzle Problem.” Nternational Journal of Computer Applications (0975 –8887), Aug. 2012. International Journal of Computer Applications (0975 –8887)Volume 51–No.22, August 2012
2. Culberson, Joseph C, and Jonathan Schaeffer. “Efficiently Searching the 15-Puzzle.” The University of Alberta, May 1994. Technical Report TR 94,08 May 1994
3. Korf, Richard E, and Larry A Taylor. “Finding Optimal Solutions to the Twenty-Four Puzzle.” Computer Science Department, University of California, Los Angeles, 1996.
4. Johnson, Storey. “ Notes on the 15'-Puzzle”. Amer. J. Math. 2(1879), 397-404, 1987.
5. Alexander Reinefeld. “Complete Solution of the Eight-Puzzle and the Benefit of Node Ordering in I D A *.” Paderborn Center for Parallel Computing Waxburger Str. 100, D-33095 Paderborn, Germany, 1993