**1. project Idea in Details:**

8-Puzzle can be described as a single agent game where we convert the start state of the puzzle into the goal state by sliding the blank tile. We first give the PEAS (Performance, Environment, Actuators and Sensors) description for the game. A suitable Performance measure is attached to the agent for it to function in an efficient manner so that it reaches the goal state in the least number of steps. This is done by attaching a suitable heuristic function which computes a value by taking the game's current state as a basis. The Environment for the game comprises of the single agent, the start state and the goal state. The Actuators define how the agent affects the environment by performing actions. The agent may or may not use its intelligence to perform one of the four actions on the blank tile namely “Slide Left”, “Slide Right”, “Slide Up” and “Slide Down”. Sensors maybe defined as how the agent gets the information from its environment. In case of 8-puzzle, information is derived from a heuristic function which provides a metric to measure the distance between the start and the goal state.

8-Puzzle represents a model problem which is very popular for measuring the performances of the heuristic search algorithms. The effort for improve search algorithms consists in determining of strong estimation heuristic functions which have to guide the search process to the most promising side of the search tree. Commonly used heuristics for this problem include counting the number of misplaced tiles (Hamming heuristic) and finding the sum of the Manhattan distances between each block and its position in the goal configuration (Manhattan heuristic).

From its inventing by Sam Loyd till now, the solving of the 8-puzzle has been represented a continuous researching subject. There can be invented many algorithms that bring a game table from any start configuration to a given goal configuration. The difficulty appears when is desired to obtain in real time an optimal solution (as number of movements to find the solution). 8-puzzle consists in 8 labeled tiles that can be moved using the only free available square space. The problem requires to be found the movements in order, starting from an initial configuration and reaching to a determined configuration. The actions represent the tiles movements, but an efficient approach from the point of view of solution search is considered that the free space is moving. In this way it result a simple problem formulation in which exist four possible actions: north movement, south movement, east movement and west movement of the free space.

----------------------------------------------------------------------

**2. Heuristic Functions:**

The heuristic algorithms take into consideration two pre-requisites: to find some distinction criteria which permit finding the solution with less resources that impose the uninformed algorithms and, on the other side, to determine the correct choices in order to reach the final state using the optimal path. So, the heuristic algorithms used information from the domain of the problem to be solved.

These information are usually used through an evaluation function which has as argument a node of the search tree and determines a number as result, indicating the measure in which the respective node is indicated for expanding.

We will use 4 heuristic functions to solve n puzzle problem, here are the heuristic functions:

1. **Number of misplaced tiles:** calculates all the tiles are out of position.
2. **Sum of Manhattan distances of the tiles from their goal positions:**

It is The most used admissible and informed heuristic function recorded so far and specified to this puzzle problem is Manhattan heuristic.

This is calculated as follows:

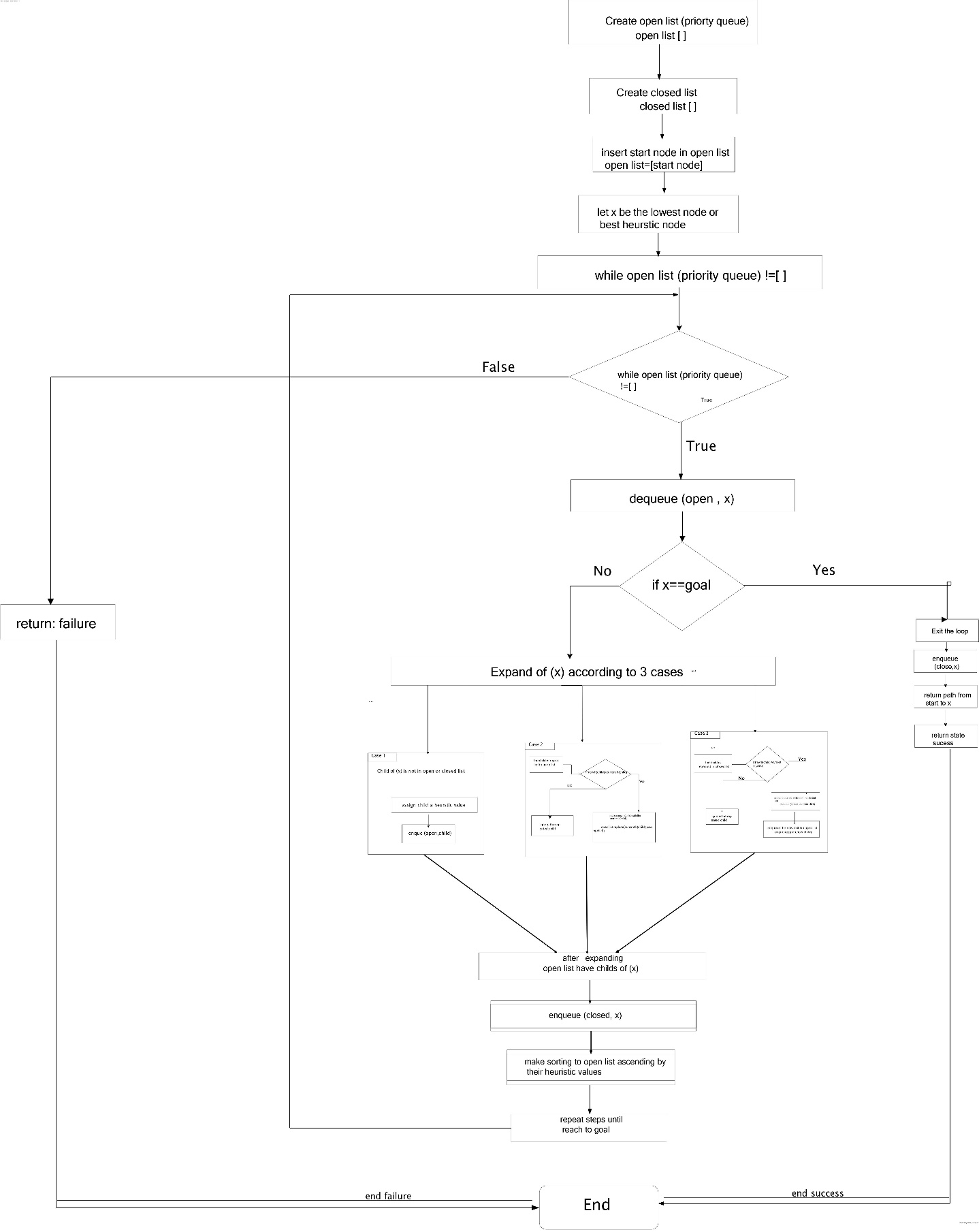
hM(S) = ∑ Manhattan Distance(k), where k{1,2,3,…,N}.

1. **Sum of Euclidean distances of the tiles from their goal positions:**

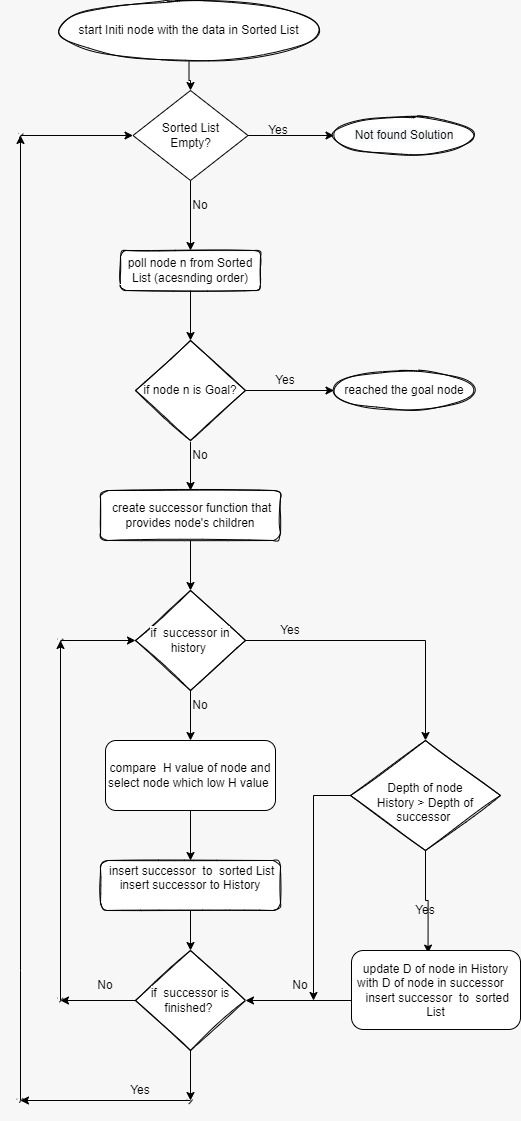
 the Euclidean distance is defined as the distance between two points. In other words, the Euclidean distance between two points

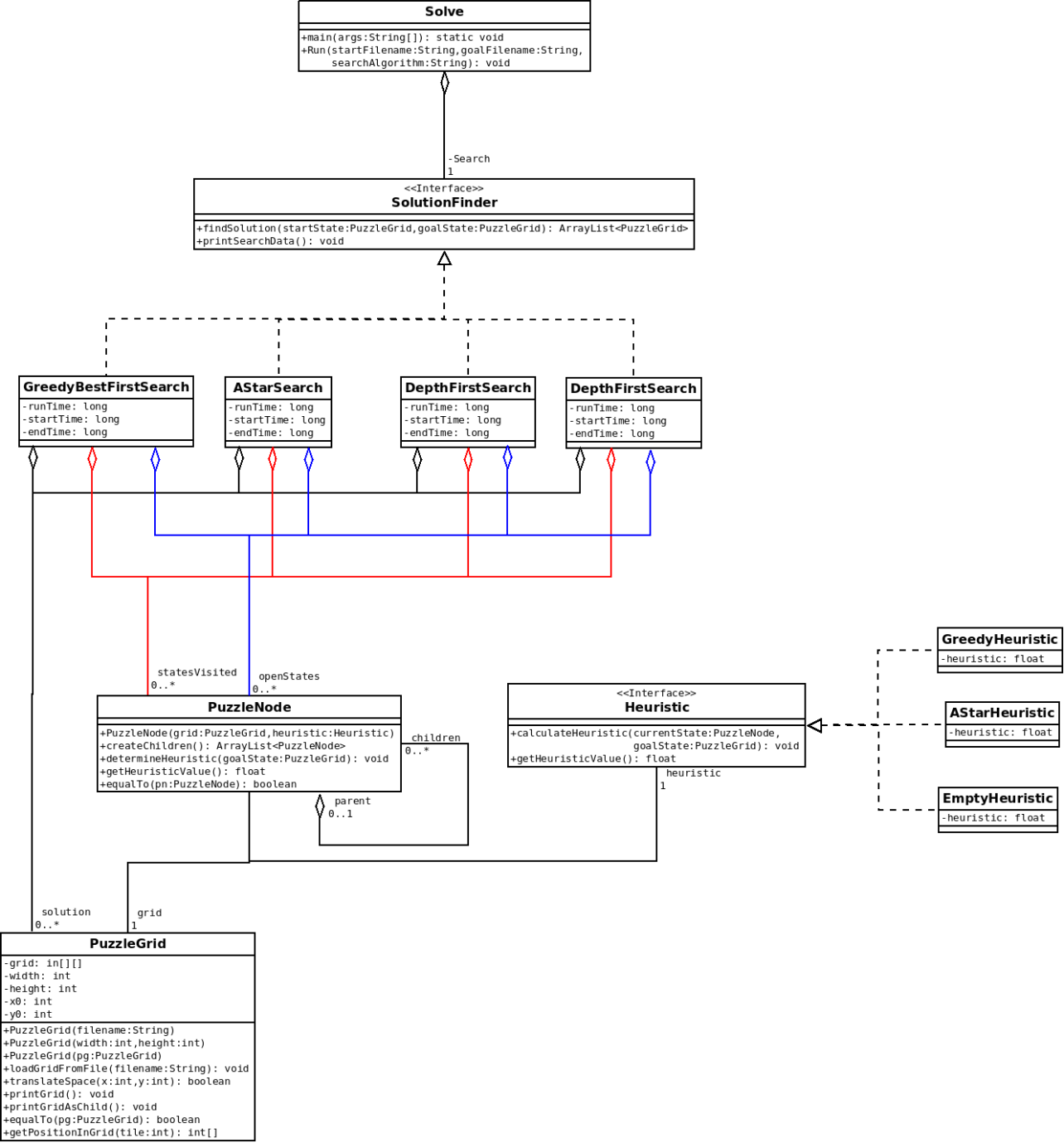
1. **Number of tiles out of row + Number of tiles out of column:** calculate all thetiles that is out of column or out of row.

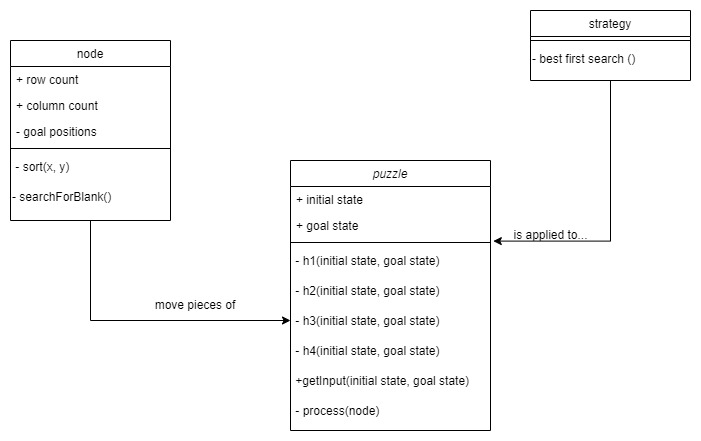
**Block diagram:**



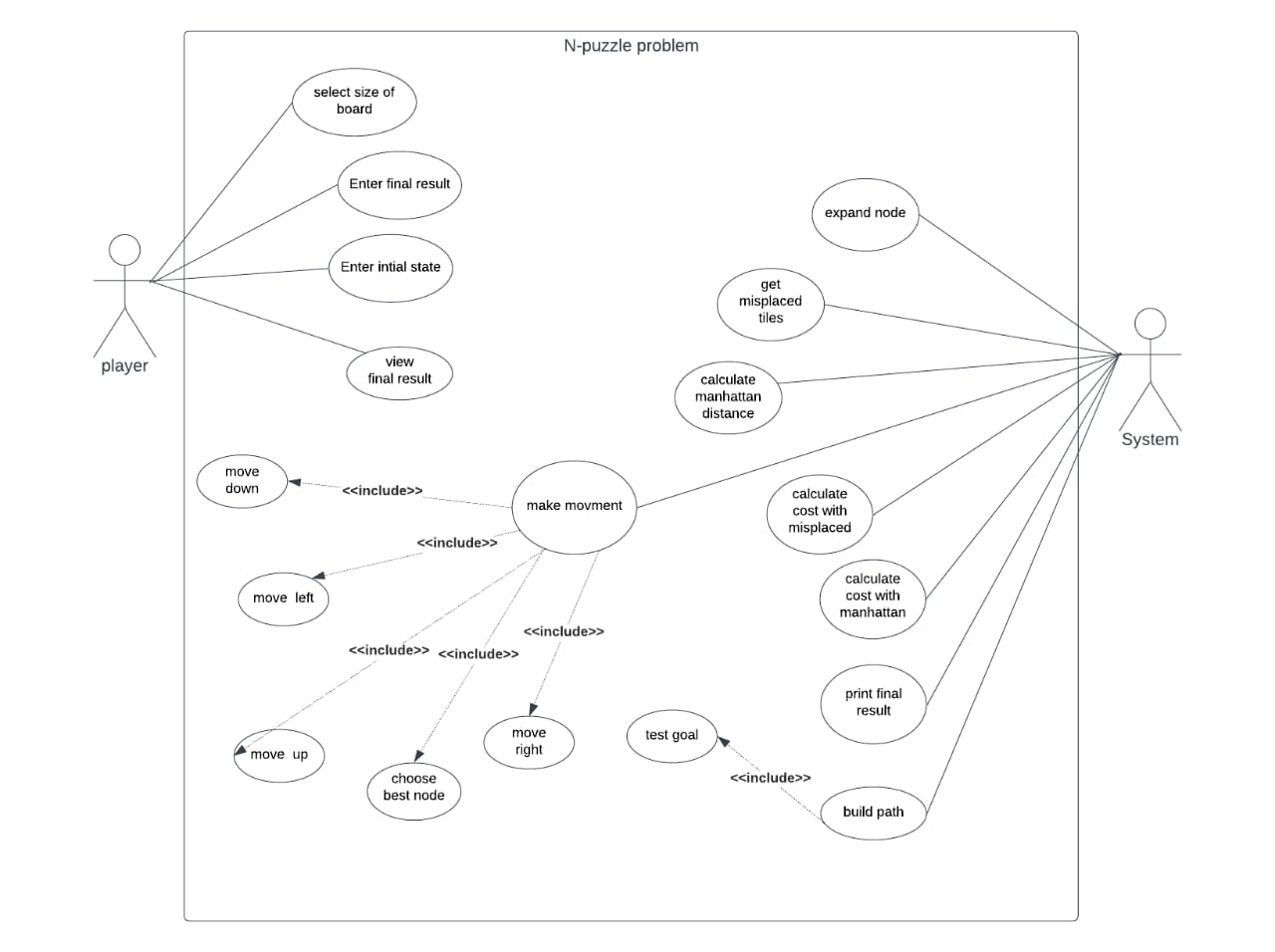
2)Data Flow Diagram



3)Class Diagram 



Use case Diagram



Similar Application in The Market

**There are many applications like this project or using the same idea and algorithm of n puzzle:**

**Sudoku *keeps your brain active and reduces the risk of Alzheimer's, a most common cause of dementia that affects a person's thinking and behavioral skills. • Stimulates your mind: The game works on your logical thinking process as you are absorbed in solving a puzzle and eventually improve your number skills. Sudoku is played on a grid of 9 x 9 spaces. Within the rows and columns are 9 “squares” (made up of 3 x 3 spaces). Each row, column and square (9 spaces each) needs to be filled out with the numbers 1-9, without repeating any numbers within the row, column or square.***

**A web crawler, or spider, *is a type of bot that is typically operated by search engines like Google and Bing. Their purpose is to index the content of websites across the Internet so that those websites can appear in search engine results.***

**A jigsaw puzzle *is a tiling puzzle that requires the assembly of often irregularly shaped interlocking and mosaiced pieces, each of which typically has a portion of a picture. When assembled, the puzzle pieces produce a complete picture.to solve this puzzle. Pick the picture of the puzzle you want to complete. Pick the number of pieces. The fewer the pieces the easier. Move the pieces to the correct spot in the puzzle.***

An initial literature review of academic publications

1)Article Name

Linear-space best-first search

2)Article Link

<https://doi.org/10.1016/0004-3702(93)90045-D>

3)Short Brief about Article

Best-first search is a general heuristic search algorithm that always expands next

frontier node of lowest cost. It includes as special cases breadth-first search, Dijkstra's single-source shortest-path algorithm, and the A∗ algorithm. Its applicability, however, is limited by its exponential memory requirement. Previous approaches to this problem, such as iterative deepening, do not expand nodes in best-first order if the cost function can decrease along a path. We present a linear-space best-first search algorithm (RBFS) that always explores new nodes in best-first order, regardless of the cost function, and expands fewer nodes than iterative deepening with a nondecreasing cost function. On the sliding-tile puzzles, RBFS with a nonmonotonic weighted evaluation function dramatically reduces computation time with only a small penalty in solution cost. In general, RBFS reduces the space complexity of best-first search from exponential to linear, at the cost of only a constant factor in time complexity in our experiments.

1)Article Name

# Pedagogical Possibilities for the N-Puzzle Problem

2)Article Link

<https://ieeexplore.ieee.org/abstract/document/4117006>

3)Short Brief about Article

In this paper we present work on a project funded by the National Science Foundation with a goal of unifying the artificial intelligence (AI) course around the theme of machine learning. Our work involves the development and testing of an adaptable framework for the presentation of core AI topics that emphasizes the relationship between AI and computer science. Several hands-on laboratory projects that can be closely integrated into an introductory AI course have been developed. We present an overview of one of the projects and describe the associated curricular materials that have been developed. The project uses machine learning as a theme to unify core AI topics in the context of the N-puzzle game. Games provide a rich framework to introduce students to search fundamentals and other core AI concepts. The paper presents several pedagogical possibilities for the N-puzzle game, the rich challenge it offers, and summarizes our experiences using it

Details of Algorithm

Best First Search Algorithm in AI

**Search algorithms** form the core of such Artificial Intelligence programs.  And while we may be inclined to think that this has limited applicability only in areas of gaming and puzzle-solving, such algorithms are in fact used in many more AI areas like route and cost optimizations, action planning, knowledge mining, robotics, autonomous driving, computational biology, software and hardware verification, theorem proving etc. In a way, many AI problems can be modelled as a search problem where the task is to reach the goal from the initial state via state transformation rules. So the search space is defined as a graph (or a tree) and the aim is to reach the goal from the initial state via the shortest path, in terms of cost, length, a combination of both

\*Concept of Algorithm

The best first search uses the concept of a priority queue and heuristic search. It is a search algorithm that works on a specific rule. The aim is to reach the goal from the initial state via the shortest path. The best [First Search algorithm in artificial intelligence](https://www.mygreatlearning.com/academy/learn-for-free/courses/best-first-search-in-artificial-intelligence?gl_blog_id=10837) is used for finding the shortest path from a given starting node to a goal node in a graph. The algorithm works by expanding the nodes of the graph in order of increasing the distance from the starting node until the goal node is reached.

\*Algorithm steps to find the solution

1. Create 2 empty lists: OPEN and CLOSED
2. Start from the initial node (say N) and put it in the ‘ordered’ OPEN list
3. Repeat the next steps until the GOAL node is reached
   1. If the OPEN list is empty, then EXIT the loop returning ‘False’
   2. Select the first/top node (say N) in the OPEN list and move it to the CLOSED list. Also, capture the information of the parent node
   3. If N is a GOAL node, then move the node to the Closed list and exit the loop returning ‘True’. The solution can be found by backtracking the path
   4. If N is not the GOAL node, expand node N to generate the ‘immediate’ next nodes linked to node N and add all those to the OPEN list
   5. Reorder the nodes in the OPEN list in ascending order according to an evaluation function f(n)

## **\*Variants of Best First Search**

The two variants of BFS are **Greedy Best First Search** and **A\* Best First Search**. Greedy BFS makes use of the Heuristic function and search and allows us to take advantage of both algorithms