8 Puzzle
AI solver using
A* and Manhattan Distance

Case Study

Submitted to:

Prof. Ria Sagum

Submitted by:

John Eris Villanueva
Kenneth Bolico

BSCS 3-2

8 Puzzle

History

The eight puzzle is a smaller version of the fifteen puzzle, which was invented in 1874 by Noyes Palmer Chapman, a postmaster in Canastota, New York. The puzzle made its way to Hartford, Connecticut, where students in the American school of the Deaf started manufacturing the puzzle and, by December 1879, selling both locally and Boston. Then the puzzle became a craze in the U.S. in February 1880.

The Board

The board is a 3x3 square that is home to eight square tiles.

The Pieces

There are eight tiles on the board, shuffled randomly from the original position.

Rules

`To move:` If there is an empty adjacent square next to a tile, a tile may be slid into the empty location.

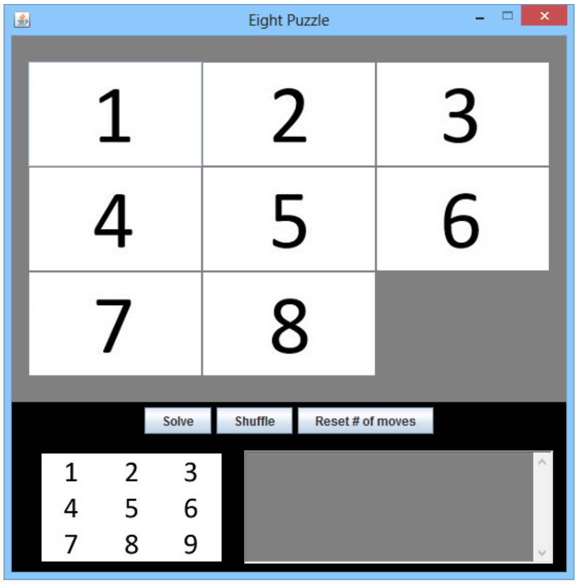
`To win:` The tiles must be moved back into their original position (goal state).

The eight puzzle is a sliding puzzle which consists of 3x3 grid of numbered squares with one square missing. The squares are jumbled when the puzzle start and the goal of this game is to unjumble the squares by only sliding a square into the empty space.

Eight Puzzle.jar

ALGORITHM: A*

HEURISTIC: Manhattan Distance



--> to run "Eight Puzzle.jar" you must have JVM installed in your computer.

--> there are 5 images available to use in the puzzle. It will be prompted at the start of the program.

--> by clicking the "Solve" button, the AI will solve the puzzle for you. The time the AI can finish solving the puzzle depends on the initial state, so there are chances that the AI can solve the puzzle quick or may take some time.

--> the "Shuffle" button randomly generates 100 moves, shuffling the puzzle.

--> you can reset the number of moves by clicking the "Reset # of moves" button.

A* and Manhattan Distance

The most widely-known form of best-first search is called **A*** search (pronounced "A-star search"). It evaluates nodes by combining g(n), the cost to reach the node, and h(n), the cost to get from the node to the goal:

$$f(n) = g(n) + h(n)$$

Since g(n) gives the path cost from the start node to node n, and h(n) is the estimated cost of the cheapest path from n to the goal, we have

$f(n)$ = estimated cost of the cheapest solution through n

Thus, if we are trying to find the cheapest solution, a reasonable thing to try first is the node with the lowest value of g(n) + h(n). It turns out that this strategy is more than just reasonable: provided that the heuristic function h(n) satisfies certain conditions, A* search is both complete and optimal.

A* is optimal if h(n) is an admissible heuristic - that is, provided that h(n) *never overestimates* the cost to reach the goal. Admissible heuristics are by nature optimistic, because they think the cost of solving the problem is less than it actually is. Since g(n) is the exact cost to reach n, we have as immediate consequence that f(n) never overestimates the true cost of a solution through n.

Manhattan Distance is the sum of the distances of the tiles from their goal positions. Because tiles cannot move along diagonals, the distance we will count is the sum of the horizontal and vertical distances. It is admissible, because all any move can do is move one tile one step closer to the goal.

for example:

current state

1	3	4
2		7
6	8	5

goal state

1	2	3
4	5	6
7	8	

$h(n) = 0 + 2 + 1 + 3 + 2 + 3 + 3 + 0 = 14$