Fortgeschrittene Algorithmen und Datenstrukturen

Seminar 7

1. Das vertrackte 8-Puzzle

8-Puzzles sind ein beliebter Rätselspaß.

Doch macht nicht nur das Herumschieben des Lochs Freude (Interessant, dass man ein Loch schieben kann, oder? Genau das passiert ja auch bei positiv dotierten Halbleitern…), sondern auch die Implementierung einer guten Lösungsstrategie!

Implementieren Sie das Rätselspiel, indem Sie die Spielzustände abstrahieren und speichern.

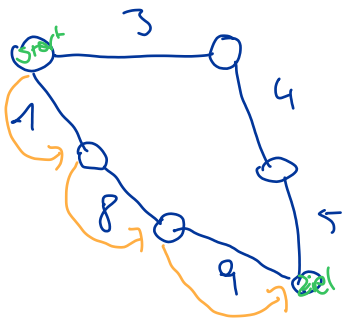
Implementieren Sie dann eine Iterative Deepening Suche, sowie eine Greedy- und eine A*-Suche mit den Heuristiken h1 (Anzahl falsch stehender Kärtchen) und h2 (~~Hamming-Distanz~~ zum Ziel), sowie einer selbst überlegten Heuristik h3 (beschreiben Sie die Heuristik) aus der Vorlesung.

Geben Sie die Anzahl der expandierten Knoten für verschiedene Suchtiefen tabellarisch (wie in den Vorlesungsfolien) aus. Vergleichen Sie die Suchen anhand der ermittelten Knotenzahlen.

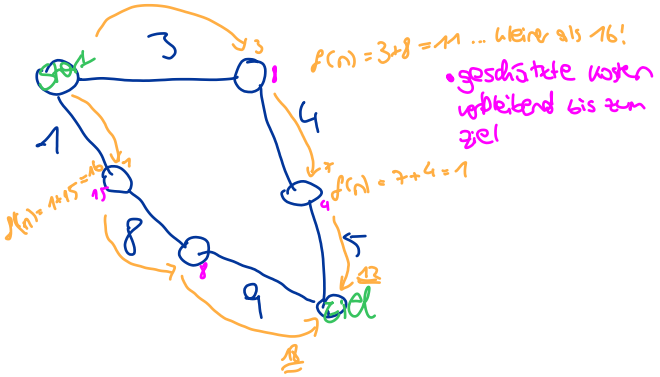
Wichtig ist, dass nicht jede zufällig erzeugte Zahlenpermutation eine Lösung hat. Es ist deshalb sinnvoll, das zu lösende Rätsel mit zufälligen Verschiebungen zu implementieren.

Heuristik h3: Anzahl der richtigen Kacheln in der Nähe des Zielorts

Greedy Search:



immer der kürzeste Weg zum Ziel



Optimales Ergebnis! A*-Suche wählt immer den kürzesten Weg.

Kosten, die durch die Heuristik geschätzt werden sind eigentlich immer geringer, als die Kosten, die tatsächlich anfallen.

f-score = h-score + g-score

A* uses a combination of heuristic value (h-score: how far the goal node is) as well as the g-score (i.e. the number of nodes traversed from the start node to current node).

In our 8-Puzzle problem, we can define the h-score as the number of misplaced tiles by comparing the current state and the goal state or summation of the Manhattan distance between misplaced nodes.
g-score will remain as the number of nodes traversed from a start node to get to the current node.

From Fig 1, we can calculate the h-score by comparing the initial(current) state and goal state and counting the number of misplaced tiles.
Thus, h-score = 5 and g-score = 0 as the number of nodes traversed from the start node to the current node is 0.

How A* solves the 8-Puzzle problem.
We first move the empty space in all the possible directions in the start state and calculate the f-score for each state.
This is called expanding the current state.
After expanding the current state, it is pushed into the closed list and the newly generated states are pushed into the open list.
A state with the least f-score is selected and expanded again. This process continues until the goal state occurs as the current state. B
asically, here we are providing the algorithm a measure to choose its actions. The algorithm chooses the best possible action and proceeds in that path.
This solves the issue of generating redundant child states, as the algorithm will expand the node with the least f-score.