AI Algorithms

Homework 1: Search Problem

Noémie Mazepa

Exercise 1: Reading

In the paper, Turing discusses several potential objections to his test for intelligence. Which of these objections still hold some significance today? Are his refutations valid? Can you think of new objections that have arisen from developments since the paper was written? Additionally, Turing predicts that by the year 2000, a computer would have a 30% chance of passing a five-minute Turing Test with an unskilled interrogator. What do you think a computer's chances would be today? Provide justification for your answers. (Approx. 1/2 page).

In his paper "Computer machinery and intelligence", Alan Turing explores the question of whether machines can think. To do so, he introduces a test called the Imitation Game to examine if a machine is capable of manifesting intelligence comparable to human intelligence. This question gathered a lot of objections like the Theological objection or the "Head-in-the-Sand" objection for example and even if those objections were raised at the time of the paper's publication, some of them still hold some significance today, when discussing intelligent machines and artificial intelligence.

First, we can consider the Mathematical objection which argues that there are certain mathematical problems that machines cannot solve and that this implies limits to their abilities and thus, intelligence. For me, this objection remains relevant today because even if they can solve a lot of different problems, machines can sometimes be unable to answer some issues, not necessarily mathematical. We can take for example some moral dilemmas like the "Trolley problem" where you must choose if you save one person or several. This type of problem solving requires empathy and a set of values which machines might not have.

Now, considering Turing's argument, he says that humans present limitations too and can also make errors. He also argues that just because one machine cannot do something doesn't mean every machine is in the same case. I do believe that being considered intelligent doesn't mean we don't ever make any mistake and that just because we are not able to solve a certain problem, it doesn't mean no one else can. In this sense, machines are similar to us. However, I do believe there are always going to be limits to machines abilities because human feelings like empathy can't really be passed onto machines via programs

Then, we can talk about Lady Lovelace's objection which assures that machines cannot create anything new because they only follow and execute provided instructions. This objection is especially relevant today as machines can generate unique creative content such as music, art, text with DALL-E or Amper Music for example. This development in content creation can raise some questions about whether a machine truly is creatively intelligent or simply modifies already existing ideas, planted in their training data. Human creativity is often based on personal experiences and emotions which machines don't have. It could therefore become important to reassess what we consider to be creative today.

Concerning Turing's refutation, he argues that while machines follow programmed instructions, they can still surprise him by giving him unexpected results. He defines creativity as an element of surprise, but we could say that "new" doesn't necessarily mean "surprising". As we said earlier, the idea of creativity today is often viewed as a result of personal experiences and emotions. This is not what Turing talks about in his argument. He focuses on how machines don't always react the way he would expect in his experiments and while I believe this can indeed be an element of surprise, it should not be mistaken for creativity. Indeed, the problem lies in the algorithms that lead the machine's behavior, rather than the machine creating something on its own.

Finally, we can mention the Theological objection. This objection claims that intelligence comes from a man's soul which machines don't have. Therefore, machines cannot be intelligent. For me, this objection remains relevant even today because as machines become more and more capable of holding a conversation, give advice in certain situations and generate new data (sound, images), there have been a lot of questions regarding whether machines can present human qualities like ethicality and awareness. It is also relevant because for the longest time, humans have believed in their superiority for having a higher intelligence and there are many concerns raised about the possibility of machines obtaining the same status.

Turing's refutation mentions that this objection limits God's powers and that if He wanted to give machines a soul, He could. The validity of this refutation really depends on our view about religion. For some, this kind of refutation could be rejected because they do not believe in God, or their religious beliefs are different from Turing's.

Over time, new objections have appeared, and we will mention two of them.

First, we will mention the Chinese Room Argument, presented by John Searle in 1980. This objection questions whether machines really understand the information they process or if they simply manipulate symbols according to given instructions, without any true understanding of the meaning. To illustrate this, he pictures himself inside a room where he follows instructions on how to respond to some Chinese characters slipped under his door. Even though he doesn't know a word of Chinese, he is still able to respond accordingly by writing the symbols as instructed. To someone on the other side of the door, Searle definitely appears to understand and write Chinese perfectly even if it's not the case at all. This situation reminds us of a computer following a program to answer questions and it raises the interrogation: do machines have a real understanding of the information they handle or are they just following instructions without any real intelligence?

Another second interesting objection we can mention is the Embodiment objection. It comes from embodied cognition theories which claim that real intelligence requires a physical body that interacts with the world. Whereas most view cognition as purely computational, embodied cognition claims that mental processes come from sensory and motor experiences. Therefore, perception, memory or even reasoning are affected by the entity's physical form and its interactions with its environment. The Embodiment objection implies that machines cannot be intelligent because they don't have a physical body from which they can have sensory and motor experiences and therefore, cannot form any sort of cognition. Even though Alan Turing's test focuses on conversation only (no physical interaction is required), this objection claims that intelligence needs physical interactions with the world which is something machines cannot do.

Finally, Turing predicts that by the year 2000, a computer would have a 30% chance of passing a five-minute Turing Test with an unskilled interrogator. Since the year 2000, machines and computers have made remarkable progress which led to significant improvements in artificial intelligence. Modern systems such as ChatGPT or Mistral are capable of generating human-like responses and can hold conversations on a lot of different topics. Modern machines could definitely exceed the 30% chance, but these computers still face some challenges especially in maintaining coherence during long interactions and sometimes demonstrating real understanding of the context. These kinds of issues could limit their performance in more rigorous testing conditions. Yet, in the context of a short interview, especially with an unskilled interrogator, a modern machine could most likely perform well. Obviously, the type of questions asked could influence the result but overall, I would estimate the chance of success at around 80%. Nevertheless, it is important to note that artificial intelligence systems still lack nuanced understanding, subtle human traits and humor that distinguish humans from machines.

Exercise 2: Search Problem

A) Formulate this puzzle as a search problem. What are the states, actions, initial state, and goal condition?

To formulate this puzzle as a search problem, we need to define: the state space, the initial space, the goal states, the actions available and a transition model

State space: a state space represents all the possible states in the environment. In this puzzle, the state space is defined by all the possible and unique configurations of a 3x3 grid that can be made by moving the number 9 up, down, left or right.

Initial state: represents the starting point of the problem. Here the initial configuration of the puzzle is the one defined in figure 1:

6	9	8
7	1	3
2	5	4

Actions: are the possible moves we can make. With this puzzle, we have four different actions: go up, go down, go left and go right. We must acknowledge that depending on the position of the number 9, some actions are not available in the moment. If we take the initial state for example, we cannot go up as we are situated on the higher edge of the grid.

Goal condition: are the desired end states we want to reach. In this problem, the goal states are all the configuration of the 3x3 grid where each row, column and diagonal sums up to 15.

Transition model: describes what each action does. In this puzzle, any action results in the number 9 exchanging its place with another adjacent number on the grid.

B) Determine whether the state space is represented as a graph or a tree.

By definition, a tree is an undirected graph where any two nodes are connected by a unique path/edge.
The main difference between a graph and a tree is that graphs can have cycles, meaning that we can revisit nodes we already encountered prior whereas trees have no cycles, there is exactly one path between any two nodes.
In our problem, no rule forbids us to go back to a state we were in a couple of actions back. We can go from one configuration to another regardless of if we have already been in this configuration. We also know that that there are multiple solutions to satisfy the condition set by the problem. There can be multiple paths between the same pair of nodes. A tree representation cannot represent everything we talked about because it has no cycle and no multiple paths.
We can then conclude that the best choice to represent the state space is a graph.

C) How large is the state space?

To determine how large the state space is, we need to know how many unique configurations there are in this problem.
Let's say that all the configurations were possible, meaning we can have any disposition of the numbers on the grid. Then, there are $9! = 9 * 8 * 7 * \ldots * 2 * 1 = 362\,880$ states possible. Indeed, we have 9 numbers that can be placed in 9 positions at the beginning. Then, one number is placed, which leaves only 8 possible positions now. After that, there are only 7 positions left, etc.

The problem is that not all configurations are actually possible. Only half of them is because of inversion. Inversion refers to if a pair of tiles are in reverse order or not, reverse order meaning the higher number appears first in the puzzle.
When we want to solve the puzzle, we can only move between configurations that have an inversion count of the same parity. A solvable state must have the same inversion count parity as the goal state. It is why only half of the configurations are possible.

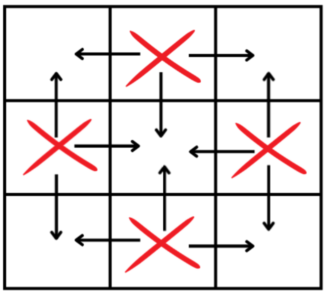
In the end there are $\frac{362\,880}{2} = 181\,440$ states possible. This is how large the state space is.

D) What is the maximum branching factor for this problem? Provide justification.

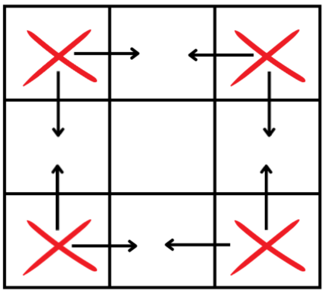
The maximum branching factor can be described as the maximum number of children a node can have in a tree or a graph. Therefore, in the context of our puzzle, we can define the maximum branching factor as the maximum number of moves we can make from a particular configuration of the puzzle.

Obviously, the number of actions that can be taken will depend on the position of number 9 on the grid. Let's look at the different possibilities:

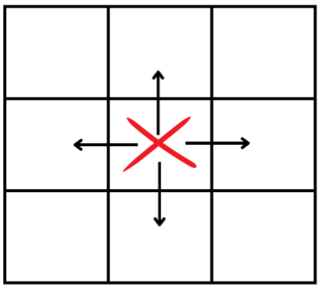
1. Number 9 is on the edge of the grid, but in the middle column or the middle row: there are 3 possible moves.



2. Number 9 is in a corner: there are 2 possible moves.



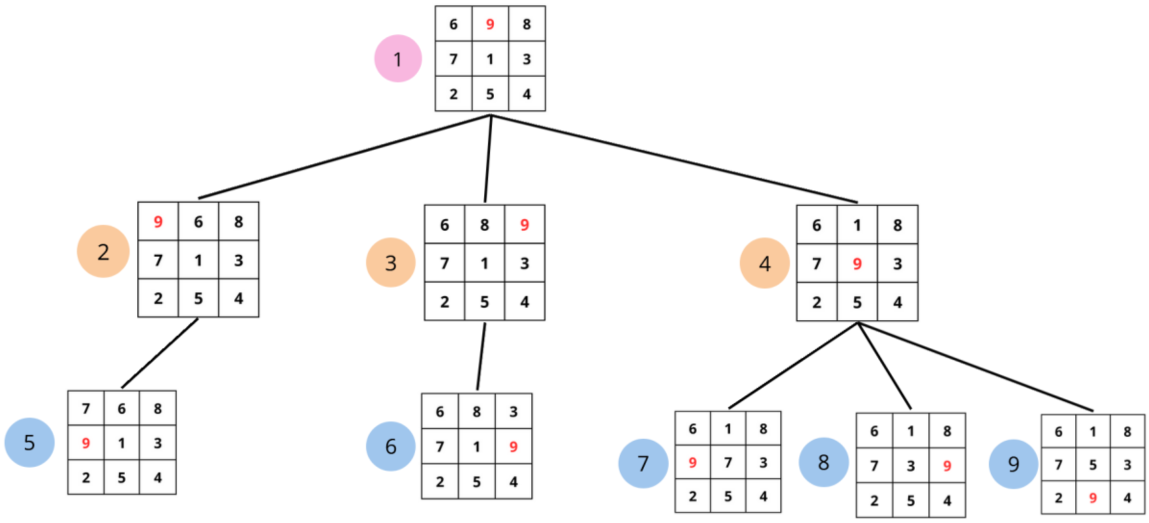
3. Finally, number 9 is in the middle of the grid: there are 4 possible moves.



These are all the possibilities for the number of moves depending on the configuration the puzzle is in. As we can see, the maximum branching factor is 4 and it occurs when number 9 is in the middle of the grid, when it can move in all four directions.

E) Draw a portion of the search graph resulting from Breadth-First Search (BFS) algorithm. Label the nodes in the order in which they are expanded.

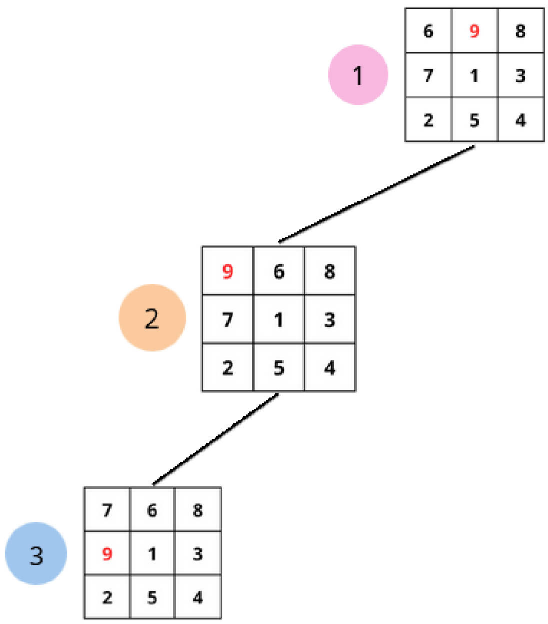
To draw a portion of the search graph resulting from the BFS algorithm, we are only going to represent up to depth=3.



We can see the order in which the nodes are expanded by the number written next to them. We can also notice that node number 9 is a solution to our problem. With BFS, we would need to expand a couple more nodes to reach the solution though. Indeed, at this moment, node number 9 is in the queue but has not yet been visited.

F) Draw a portion of the graph search generated by the Depth-First Search (DFS) algorithm and label the states in the order they are expanded.

To draw a portion of the search graph resulting from the DFS algorithm, we are only going to represent up to depth=3.

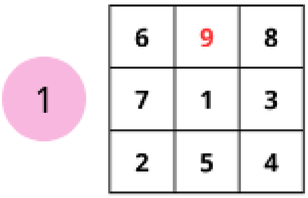


With the DFS algorithm, we only have three nodes because it needs to go to maximal depth of that branch in order to expand the other children of the root node. Because, we stopped at depth=3, the algorithm didn't reach maximal depth.

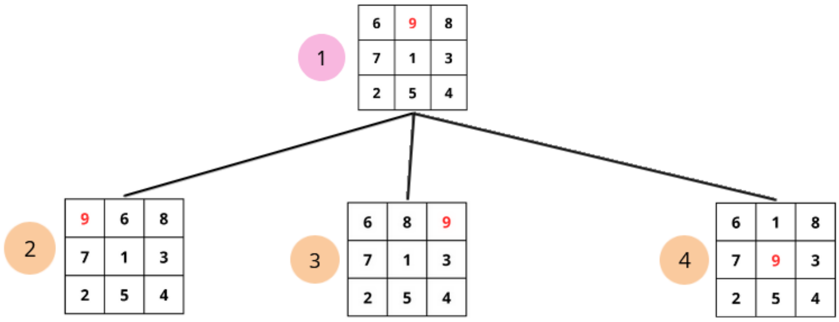
G) Draw a portion of the graph search generated by the Iterative Deepening Search (IDS) algorithm and label the order in which each state is expanded.

For the IDS algorithm, we are going to represent the graph search for different depths: 0, 1 and 2.

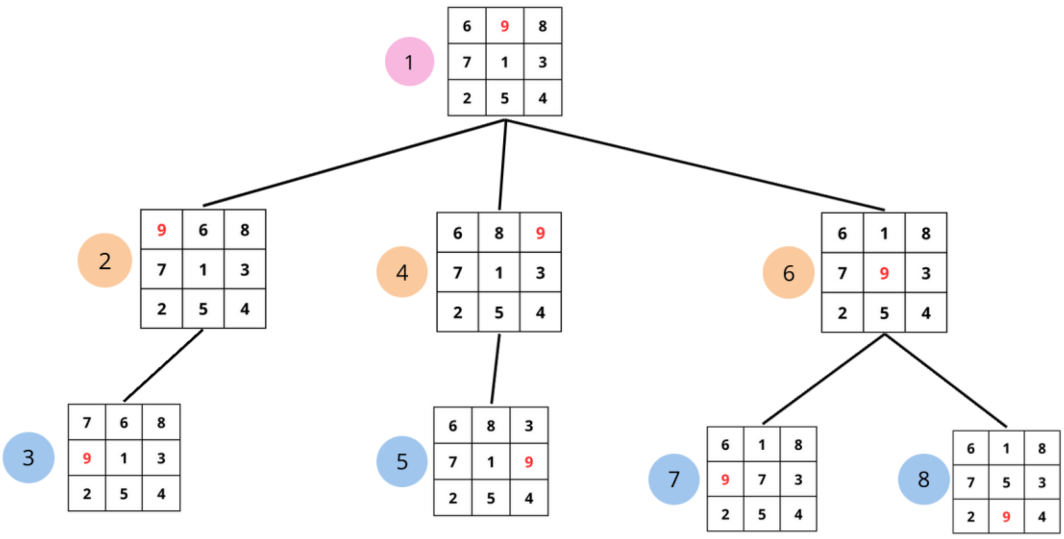
For depth=0:



For depth=1:



For depth=2:



Here, we can stop at node 8 because we reached a solution. Unlike with the BFS, we don't expand to a 9th node because here, we perform a DFS. All the other nodes (3,5 and 7) have been fully expanded until depth=2 and when we expand node 8, there are no other nodes in the frontier.

H) What are the advantages and disadvantages of Breadth-First Search and Iterative Deepening Search for this problem? Would Depth-First Search with no limit be a better or worse approach? Why ?

Breadth-First Search is an algorithm where you first look wide before you look deep. It explores the graph level by level. Nodes are expanded in the same order in which they are generated, and the frontier is a First-In, First-Out queue.

Some advantages of BFS for this problem are:

Completeness: the BFS will always find a solution if one exists without getting stuck or missing a solution. Because BFS explores all the nodes, level-by-level, it is sure to eventually find a solution (if it exists). Since our problem has a finite state space, BFS will definitely find a solution at some point.

Optimality: if the cost path is the same for each action, which is the case with our problem, the BFS algorithm is optimal. It will always find a solution with the fewest moves possible because it explores the shallowest nodes first. It is an advantage in the context of our problem because we would like to minimize the number of moves it takes to find one satisfying condition.

Some disadvantages of BFS are:

Space complexity: because it expands all the nodes at each level, the search space can quickly become huge and complex. Indeed, the frontier can easily take a lot of space with a complexity up to O(b^d) with d being the maximal depth and b being the number of nodes. The number of nodes stored grows exponentially with the depth. In the context of our puzzle, it definitely is a disadvantage because it needs to remember all of the nodes expanded but not yet visited and we saw in an earlier question that there are a lot of different configurations.

Time complexity: same as with space complexity, because all of the nodes are expanded at each level, the time required to search for a solution can grow quickly if the tree/graph is quite deep and complex. Time complexity is similar to space complexity as it is also O(b^d). For the puzzle, if the correct configuration is situated quite deep in the search space, it will take a lot of time to explore all the possible paths.

Iterative Deepening Search is a form of depth limited search. It basically runs DFS several times and we progressively increase the depth at each iteration. The nodes at the maximal depth are also treated as if they had no successors.

Some advantages of IDS for this problem are:

Completeness: just like BFS, the IDS algorithm is complete and will always find a solution if one exists. It is an advantage in the context of our puzzle because there are different configurations that can be a solution to the problem. Because IDS is complete, we are guaranteed to find at least one of these solutions.

Optimality: IDS is also optimal when all actions have the same cost path like in our problem. It is an advantage because the goal often is to find a working solution in the fewest moves possible.

Space efficiency: IDS only keeps track of the nodes in the current branch, not the entirety of the nodes on the same level like BFS. The number of nodes stored is linear in function of depth. For our puzzle, where the search space is quite large, the space will be much more manageable even for deep searches.

Some disadvantages of IDS for this problem are:

Redundancy: because it repeats the search several times while increasing the depth, IDS will most likely repeat the same expansions for the smallest depths. This means that, with our problem, the IDS will encounter the same first few configurations each time it increases the depth. The redundancy isn't so bad if the iterations are few but if the first solution is encountered in the deeper parts of the search space, then there will be a lot of repetitions.

Time complexity: as a result of redundancy, the IDS will have an important time complexity. The repetition of DFS algorithms at different depth slows down the search and increases the computational time.

Depth-First Search with no limit would most likely be a worse approach than BFS or IDS for our problem. On the one hand, DFS is not optimal because it finds the first solution it encounters, without considering the depth or cost. Therefore, there is no guarantee it will be the optimal one. On the other hand, with no limit, the depth could be infinite and DFS could get stuck in cycles and never find a solution, meaning that it is only complete in finite spaces. In conclusion, DFS is definitely not the better approach to solve our problem.

I) Propose a non-trivial admissible heuristic for solving this problem.

A heuristic is a function that evaluates the cost of the cheapest path from the current node to the goal state. They are used to make the search process more efficient by guiding said search towards the solution, while prioritizing actions that cost less.
If a heuristic is admissible, it means that, for every node, it never overestimates the cost of reaching the goal state from that node. Thus, we have $\forall n, h(n) \leq actual\_cost(n)$ with n being a node.
Obviously, to estimate the cost path to reach the state goal, the heuristic needs to know the goal state, or in our case, one of the goal states.

To solve this problem, we could use the Manhattan Distance heuristic, a very well-known heuristic often used in search problems where we need to rearrange elements to

correspond to a particular configuration. The goal of this heuristic is to sum up the Manhattan distance of each element of the puzzle from its position in the goal state. To calculate the Manhattan distance of one tile, we need to add the horizontal distance (number of moves horizontally) to the vertical distance (number of moves vertically) needed to reach the goal position. For the entire puzzle, we just sum up all the singular Manhattan distances.
Because this heuristic counts the minimum number of moves needed for a tile to reach its goal position, it never overestimates the cost. It doesn't consider the potential additional moves required to move that tile with number 9, it just assumes number 9 is able to swap directly with the tile. Therefore, this heuristic can only underestimate or exactly estimate the cost. It is why it is admissible.

We could also have proposed the Misplaced Tiles heuristic but its estimation of cost, although admissible, is much less precise than Manhattan Distance and will eventually lead to more nodes generated. In conclusion, Manhattan Distance is a much more precise estimate of cost than Misplaced Tiles, therefore helping us make a more informed and efficient search.

Bibliography

(N.d.). *Internet encyclopedia of philosophy*. Retrieved October 19, 2024, from <https://iep.utm.edu/chinese-room-argument/>

Cole, D. (2020). The Chinese Room Argument. *Stanford Encyclopedia of Philosophy*. Stanford University. Retrieved October 19, 2024, from <https://plato.stanford.edu/entries/chinese-room/>

Embodied cognition. (2024). *Wikipedia*. Wikimedia Foundation. Retrieved October 19, 2024, from <https://en.wikipedia.org/wiki/Embodied_cognition>

Shapiro, L., & Spaulding, S. (2021). Embodied cognition. *Stanford Encyclopedia of Philosophy*. Stanford University. Retrieved October 19, 2024, from <https://plato.stanford.edu/entries/embodied-cognition/#ThreThemEmboCogn>

GeeksforGeeks. (2022). How to check if an instance of 8 puzzle is solvable? GeeksforGeeks. GeeksforGeeks. Retrieved October 19, 2024, from <https://www.geeksforgeeks.org/check-instance-8-puzzle-solvable/>

15 puzzle. (2024a). Wikipedia. Wikimedia Foundation. Retrieved October 19, 2024, from <https://en.wikipedia.org/wiki/15_puzzle>