Introducion to Artificial Intelligence

Spring 2019

Miniproject 1: Kuromasu

Carolin Brunn, Mateusz Ruszczyk

Pages: 4

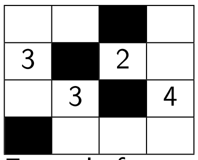
15.04.2019

Kuromasu – Solving puzzle using depth first search and A* algorithm

Rules of the puzzle

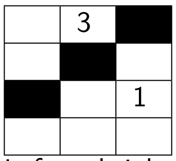
The board of a Kuromasu game is a rectangular grid, which consists of single cells. Some of the cells contain numbers. During the game the cells can be coloured with black or white. The fields, which contain numbers, are always white. Fields that are not decided yet are grey.

The goal of the game is to insert black fields so that the number of reachable white fields from the fields containing numbers equals the number written into the field. The number field itself is also counted. A white field can be reached if it is possible to go straight from the number field to the reachable field in a vertical or horizontal line. So they are in the same row or column as the number field without any interruption in between. It is not possible to skip black fields, so a black field is an interruption.



Example for a solved board

Additionally all white fields must be connected, so that it is possible to go from each white field to each other white field by using only vertical or horizontal steps, not diagonal. Black fields may not be connected vertically or horizontally, so they do not share a common side. They may be connected diagonally. Nevertheless as a conclusion of the previous rules it results that it cannot be possible to go in a circle or from the edge to the board again to the edge by doing only diagonal steps using black fields. If that was the case, then one part of the white fields would be separated from the rest.



Left and right half of the board are separated by black

Puzzle space/search space

The puzzle space contains all the descriptions of possible combinations of colouration of the board. It also contains the combinations which do not fulfill the constraints of the rules of the game. In our case, one description of the board is an array with entries for each cell, specifying its colour. Possible entries are white, black, not decided or the number written into the cell.

The search space contains only the valid states. That means all the colourations of the board which match the rules which were specified above. For example, states with adjacent black fields are not taken into account. A state with only white and number fields is considered because it is valid although it is not yet solved.

Cost and Heuristic function

To implement the A* algorithm we had to calculate an objective function g(x) + h(x), with g(x) as the cost function and h(x) as a heuristic function. As a cost function we counted how many black fields were already inserted into the board.
As our heuristic function we calculated 1 minus the quotient of the sum of white fields that were reachable outgoing from number fields over the sum of the numbers that were written into number fields. Reachable fields are all the white fields that are in the same column or row as the number field without any black fields in between. Again the number fields themselves are counted.

3		
		1

So in this case the number of all reachable white fields would be $5 + 5 = 10$. The sum of the numbers written into number fields is $3 + 1 = 4$. Therefore for this example our heuristic function is $h(x) = 1 - \frac{reachablefields}{\sum numbers} = 1 - \frac{5+5}{3+1} = -\frac{6}{4}$.

Our goal is that the result of the heuristic function becomes zero.

At first the result of the heuristic function will be ≤ 0. If it is zero that means that the board is a possible solution. So most likely the result will be negative in the beginning. If the result of the heuristic function is positive it means that the board is not longer solvable, because proceeding from this field we are not quitting black fields again. Thus, there is no way that the number of reachable fields increases and the heuristic function could decrease again.

This heuristic function is monotonic because we are adding black fields to approach the solved board. Every time a black field is added the number of reachable fields stays either the same or decreases. Once we insert a new black field, that means we increase g(x), we are sure that no board with a smaller cost function is a solution. This is because we are looking at all possible children before deciding to which we want to go. So the number of reachable fields cannot increase again and therefore the heuristic cannot decrease.

3	■	
		■
■		2

Board is not solvable anymore by inserting black cells.

In this example the number of reachble fields equals 4 but the sum of number fields is 5. Thus, our heuristic function is $h(x) = 1 - \frac{reachablefields}{\sum numbers} = 1 - \frac{2+2}{3+2} = \frac{1}{5}$. The result is positive and as we can see the board is not solvable by inserting black fields.

Performance analysis

Overall the performance of the A* algorithm was better than the performance of the depth first search.
For this 3x3 board the number of iterations for DFS were 23 with an execution time of 0.011s

and for A* 7 in 0.003s. For another 3x3 board the difference was even more significant. DFS took 53 iteration steps in 0.02s and A* only 9 steps in 0.004s.

	4	3
		2
3	4	

23 - 7

	3	
3		2
	2	

53 - 9

For the 4x4 board the difference was getting more relevant. The DFS needed about 10 times more iterations than A*. For the two examples DFS needed 1067 (0.49s) and 1015 (0.46) iterations and the A* only 96 and 74, with iteration times of 0.056s and 0.046s.

		4	
3			
			2
	7		

1067 - 96

		5	
			2
5			
	5		

1015 - 74

Still one of the biggest differences we could observe was a 5x5 board, where the DFS algorithm needed almost 20 times more iterations than A*. The difference was between 22987 and 123 iterations. The execution time for DFS was 13.97s and for A* 0.09s

			6	
			8	7
4	6			
	8			

22987 - 123

	3			
				4
		9		
3				
			8	

32037 - 440

Nevertheless we also could observe one example for which the performance of A* was worse tan for DFS. A* needed 750 iterations in 0.38s and DFS only 565 in 0.22s.

2			
		3	
	5		
			6

565 - 750

Nonetheless this difference is not as big as the differences we could in the examples where A* performed better. We tried out some more boards which are not shown here and the results were always similar.
We can therefore conclude that the performance was increased by a high factor using A* instead of DFS.