Introduction to Artificial Intelligence: Assignment 3 - Using the A* Algorithm

Andreas Dørum, Martin Baklid

October 2015

0.1 Introduction

We have written one program that solves all subtasks in A. The program can parse both boards with obstacles only, and boards with different cell costs. With a command-line argument, the user can specify which algorithm should be used to solve the board. ”b” will instruct the program to use Breadth-First-Search (BFS), ”d” will use Dijkstra’s Algorithm, and any other or none arguments will use the A* algorithm.

Chapter 1

Subproblem A.1

Our A* implementation is attached in files. Astar.py, contains the search algorithm. prique.py, contains a priority queue implementation for A* and a first-in first-out (FIFO)-queue for use with BFS. Pathfind_problem.py contains the problem specification, and heuristic logic for solving the 2D pathfinding problem. Search_node.py contains the SearchNode class. Board_image_gen.py contains the image generator, a class that uses the Pillow library to generate visualizations.

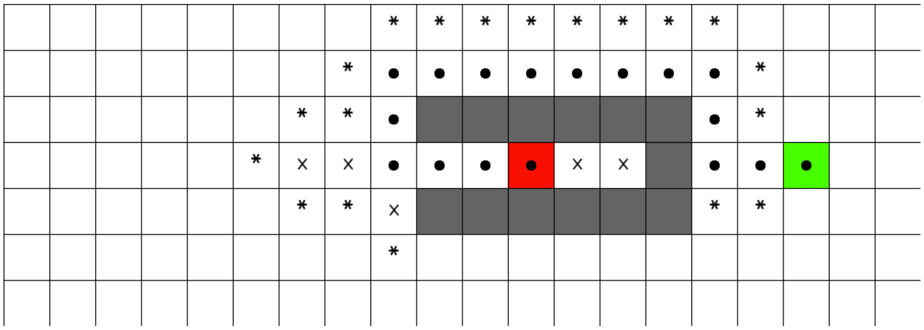


Figure 1.1: A* executed on board-1-1. Dots represents the best path A* calculated, x represents a closed node and * represents an open node, or a node in the frontier

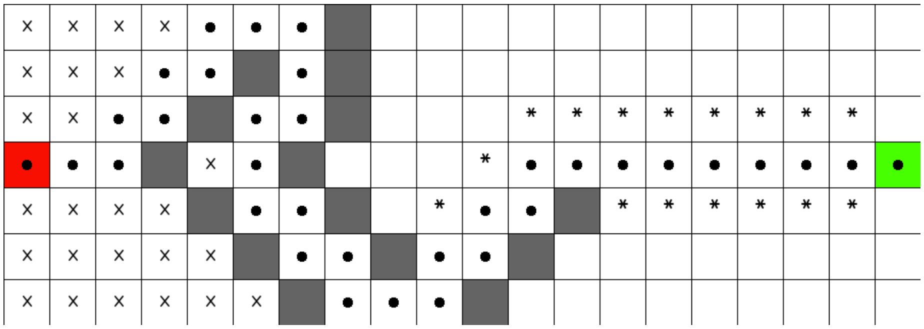


Figure 1.2: A* executed on board-1-2. Dots represents the best path A* calculated, x represents a closed node and * represents an open node, or a node in the frontier

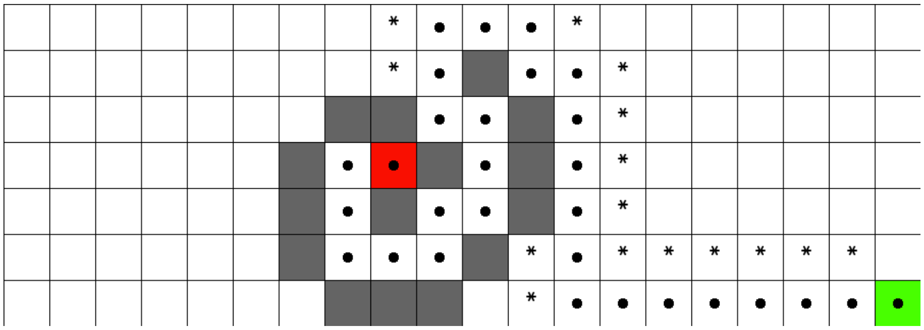


Figure 1.3: A* executed on board-1-3. Dots represents the best path A* calculated, x represents a closed node and * represents an open node, or a node in the frontier

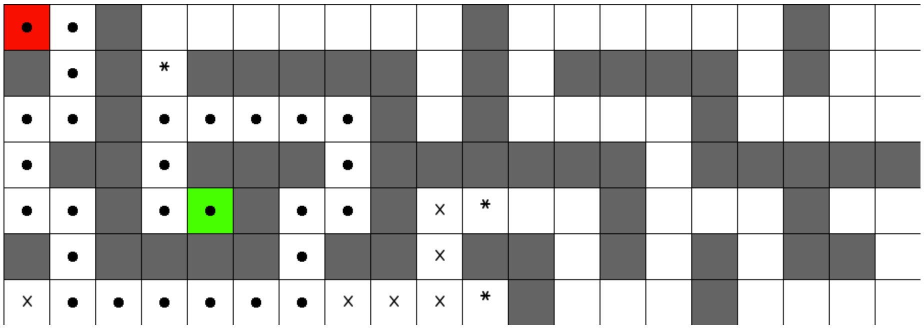


Figure 1.4: A* executed on board-1-4. Dots represents the best path A* calculated, x represents a closed node and * represents an open node, or a node in the frontier

Chapter 2

Subproblem A.2

To make our A* implementation to work on a graph with diffrent cellcosts we only needed to change the arc_cost function

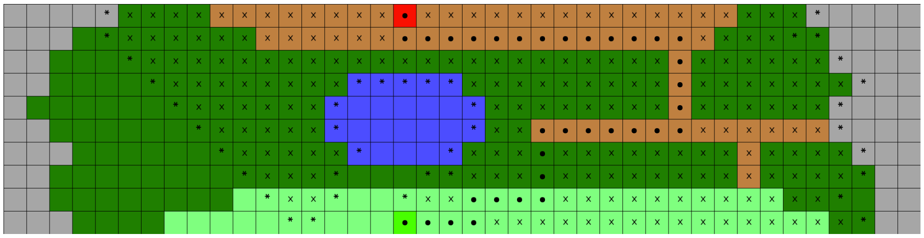


Figure 2.1: A* executed on board-2-1. Dots represents the best path A* calculated, x represents a closed node and * represents an open node, or a node in the frontier

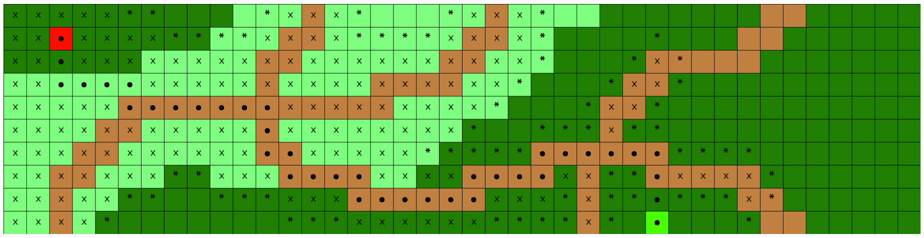


Figure 2.2: A* executed on board-2-2. Dots represents the best path A* calculated, x represents a closed node and * represents an open node, or a node in the frontier

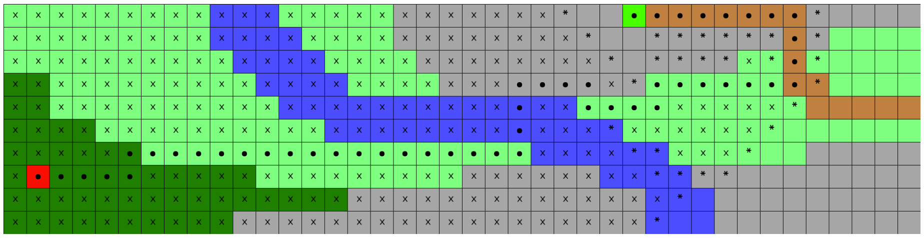


Figure 2.3: A* executed on board-2-3. Dots represents the best path A* calculated, x represents a closed node and * represents an open node, or a node in the frontier

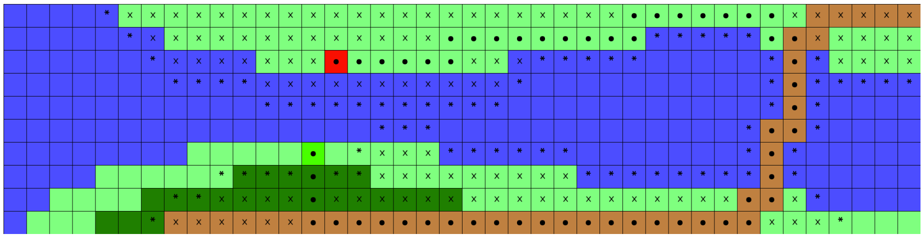


Figure 2.4: A* executed on board-2-4. Dots represents the best path A* calculated, x represents a closed node and * represents an open node, or a node in the frontier

Chapter 3

Subproblem A.3

In this subproblem we compared the performance of three different algorithms in solving pathfinding problems: BFS, Dijkstra, and A*. In the attached document, stat.txt, we have created a document showing the length of the path, open nodes and closed nodes for the different algorithms on the different boards. We also compare the visual representations of the executions of the algorithms.

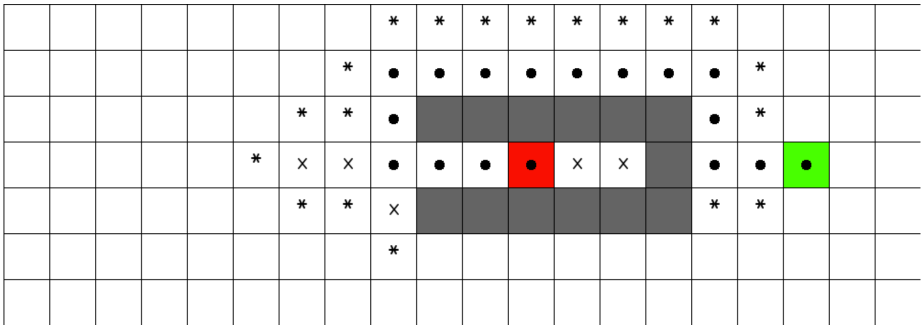
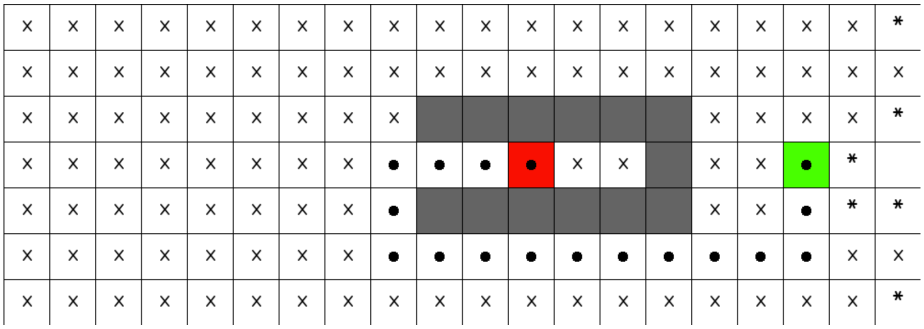


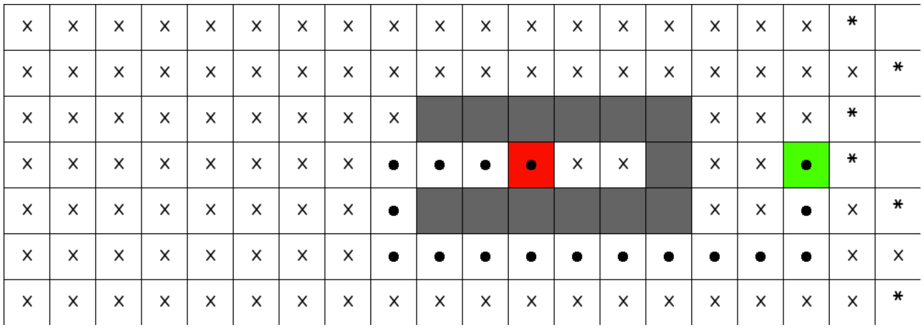
Figure 3.1: A* executed on board-1-1. Dots represents the best path A* calculated, x represents a closed node and * represents an open node, or a node in the frontier

Comparison of Board 1-1 solutions: The most notable difference is the fewer calculations done by A* than by the two other algorithms. Djikstra and BFS expands nearly every possible node on the board. The different route taken, above and below the obstacles, is merely a coincidence. Depending on implementation, bfs and dijkstra could have given the same results, but because of the sorting method used when generating the images, they give slightly different results in all the images.



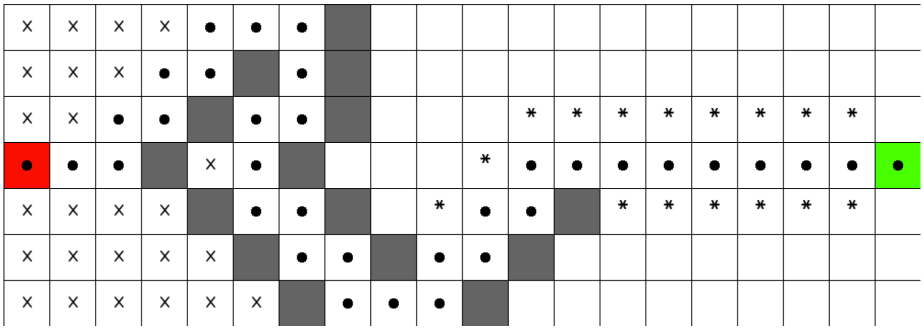
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	*
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
x	x	x	x	x	x	x	x	x	▨	▨	▨	▨	▨	▨	x	x	x	x	*
x	x	x	x	x	x	x	x	•	•	•	🟥•	x	x	▨	x	x	🟩•	*	
x	x	x	x	x	x	x	x	•	▨	▨	▨	▨	▨	▨	x	x	•	*	*
x	x	x	x	x	x	x	x	•	•	•	•	•	•	•	•	•	•	x	x
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	*

Figure 3.2: Dijkstra executed on board-1-1. Dots represents the best path dijkstra calculated, x represents a closed node and * represents an open node, or a node in the frontier



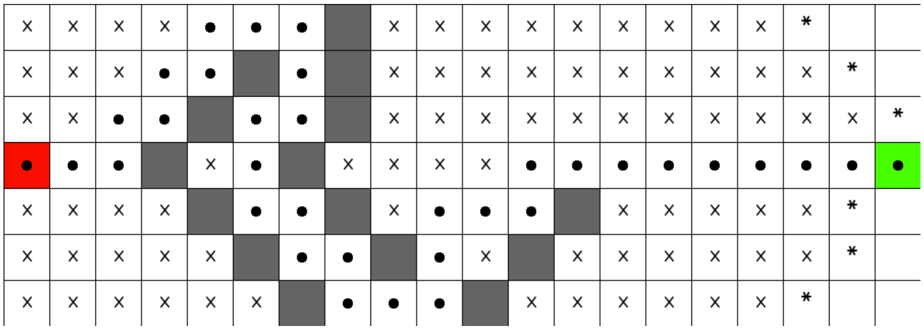
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	*	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	*
x	x	x	x	x	x	x	x	x	▨	▨	▨	▨	▨	▨	x	x	x	*	
x	x	x	x	x	x	x	x	•	•	•	🟥•	x	x	▨	x	x	🟩•	*	
x	x	x	x	x	x	x	x	•	▨	▨	▨	▨	▨	▨	x	x	•	x	*
x	x	x	x	x	x	x	x	•	•	•	•	•	•	•	•	•	•	x	x
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	*

Figure 3.3: Breadth first search executed on board-1-1. Dots represents the best path breadth first search calculated, x represents a closed node and * represents an open node, or a node in the frontier



x	x	x	x	●	●	●	▨												
x	x	x	●	●	▨	●	▨												
x	x	●	●	▨	●	●	▨				*	*	*	*	*	*	*	*	
●	●	●	▨	x	●	▨				*	●	●	●	●	●	●	●	●	●
x	x	x	x	▨	●	●	▨		*	●	●	▨	*	*	*	*	*	*	
x	x	x	x	x	▨	●	●	▨	●	●	▨								
x	x	x	x	x	x	▨	●	●	●	▨									

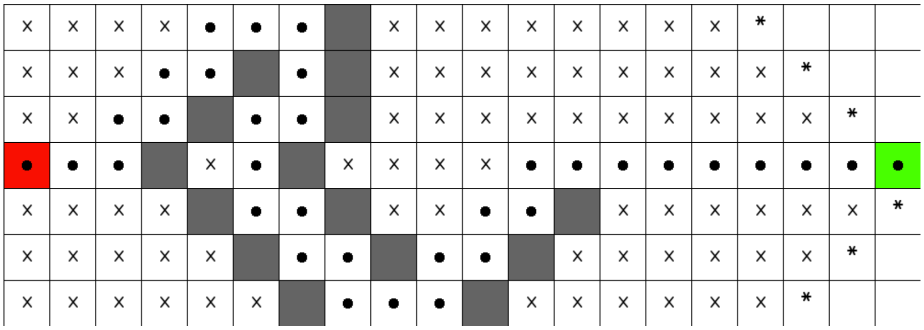
Figure 3.4: A* executed on board-1-2. Dots represents the best path A* calculated, x represents a closed node and * represents an open node, or a node in the frontier



x	x	x	x	●	●	●	▨	x	x	x	x	x	x	x	x	x	*		
x	x	x	●	●	▨	●	▨	x	x	x	x	x	x	x	x	x	x	*	
x	x	●	●	▨	●	●	▨	x	x	x	x	x	x	x	x	x	x	x	*
●	●	●	▨	x	●	▨	x	x	x	x	●	●	●	●	●	●	●	●	●
x	x	x	x	▨	●	●	▨	x	●	●	●	▨	x	x	x	x	x	*	
x	x	x	x	x	▨	●	●	▨	●	x	▨	x	x	x	x	x	x	*	
x	x	x	x	x	x	▨	●	●	●	▨	x	x	x	x	x	x	*		

Figure 3.5: Dijkstra executed on board-1-2. Dots represents the best path dijkstra calculated, x represents a closed node and * represents an open node, or a node in the frontier

Comparison of Board 1-2 solutions: As before, the A* algorithms expands fewer nodes than the uninformed algorithms.



x	x	x	x	●	●	●	▨	x	x	x	x	x	x	x	x	*			
x	x	x	●	●	▨	●	▨	x	x	x	x	x	x	x	x	x	*		
x	x	●	●	▨	●	●	▨	x	x	x	x	x	x	x	x	x	x	*	
🟥 ●	●	●	▨	x	●	▨	x	x	x	x	●	●	●	●	●	●	●	●	🟩 ●
x	x	x	x	▨	●	●	▨	x	x	●	●	▨	x	x	x	x	x	x	*
x	x	x	x	x	▨	●	●	▨	●	●	▨	x	x	x	x	x	x	*	
x	x	x	x	x	x	▨	●	●	●	▨	x	x	x	x	x	x	*		

Figure 3.6: Breadth first search executed on board-1-2. Dots represents the best path breadth first search calculated, x represents a closed node and * represents an open node, or a node in the frontier

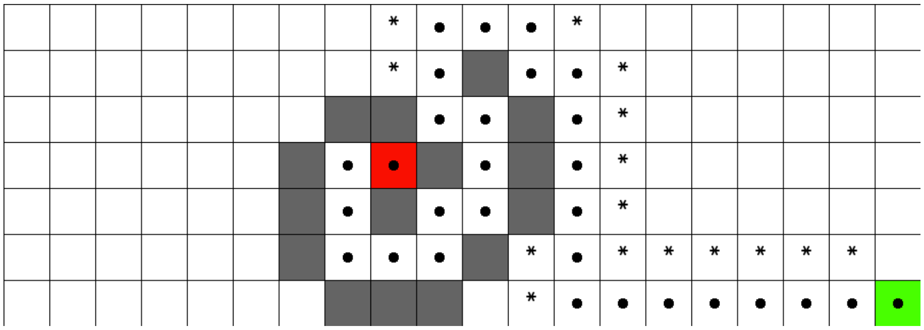


Figure 3.7: A* executed on board-1-3. Dots represents the best path A* calculated, x represents a closed node and * represents an open node, or a node in the frontier

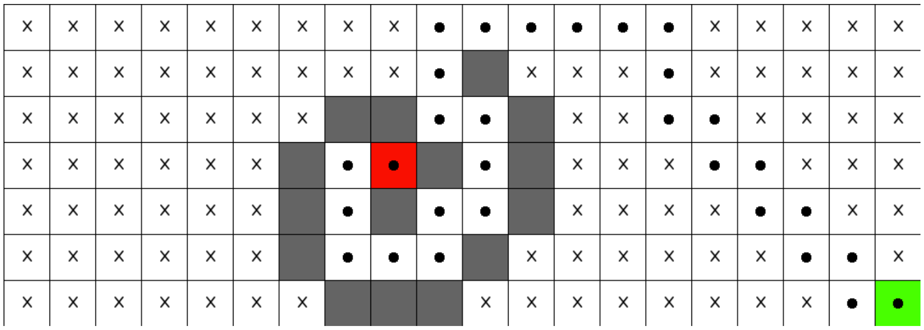
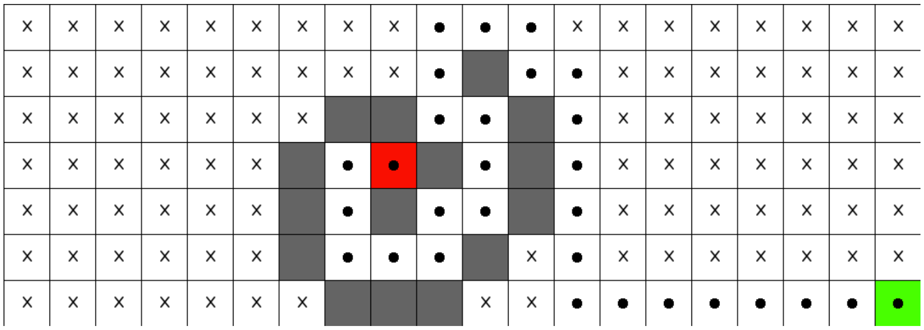


Figure 3.8: Dijkstra executed on board-1-3. Dots represents the best path dijkstra calculated, x represents a closed node and * represents an open node, or a node in the frontier

Comparison of Board 1-3 solutions: In this case, the efficency of the A* algorithm is particularly evident. While the uninformed algorithms searches the entire left hand side of the board, A* quickly decides on searching on the right hand side only. It can do this, because it's heuristic gives it a sense of direction towards the goal, which makes it prefer moving to the right.



x	x	x	x	x	x	x	x	x	●	●	●	x	x	x	x	x	x	x	x
x	x	x	x	x	x	x	x	x	●	▨	●	●	x	x	x	x	x	x	x
x	x	x	x	x	x	x	▨	▨	●	●	▨	●	x	x	x	x	x	x	x
x	x	x	x	x	x	▨	●	🟥●	▨	●	▨	●	x	x	x	x	x	x	x
x	x	x	x	x	x	▨	●	▨	●	●	▨	●	x	x	x	x	x	x	x
x	x	x	x	x	x	▨	●	●	●	▨	x	●	x	x	x	x	x	x	x
x	x	x	x	x	x	x	▨	▨	▨	x	x	●	●	●	●	●	●	●	🟩●

Figure 3.9: Breadth first search executed on board-1-3. Dots represents the best path breadth first search calculated, x represents a closed node and * represents an open node, or a node in the frontier

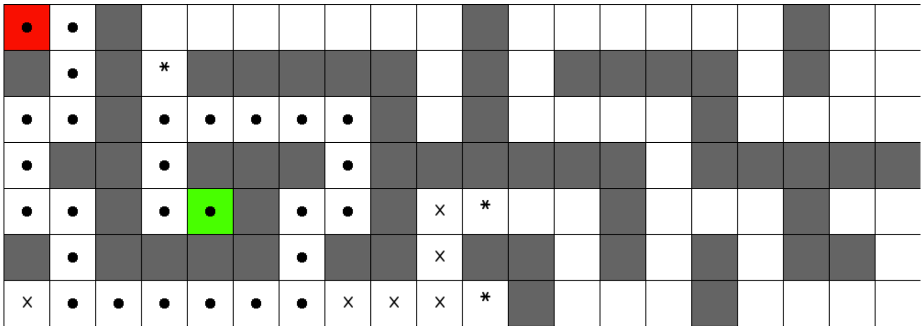


Figure 3.10: A* executed on board-1-4. Dots represents the best path A* calculated, x represents a closed node and * represents an open node, or a node in the frontier

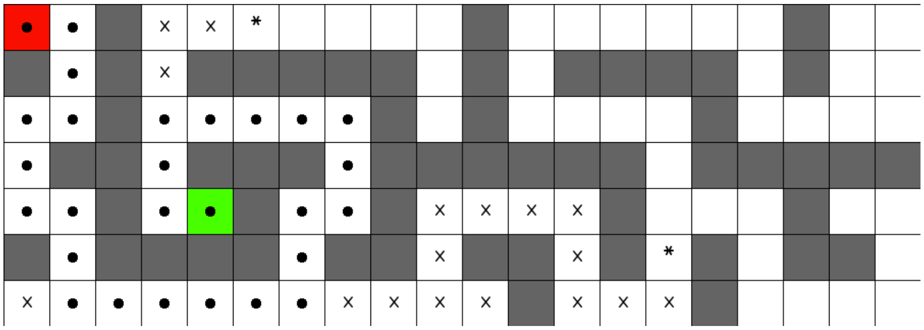


Figure 3.11: Dijkstra executed on board-1-4. Dots represents the best path dijkstra calculated, x represents a closed node and * represents an open node, or a node in the frontier

Comparison of Board 1-4 solutions: In this case, A* is only slightly better than the other algorithms. This is because of the limited possibility space. The maze-like structure of the boards lends itself to the "flooding" method.

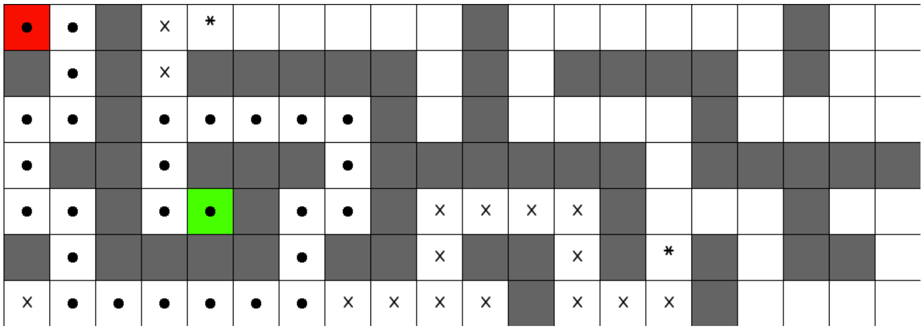


Figure 3.12: Breadth first search executed on board-1-4. Dots represents the best path breadth first search calculated, x represents a closed node and * represents an open node, or a node in the frontier

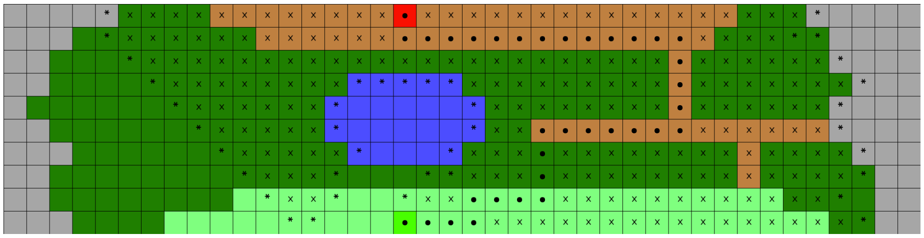


Figure 3.13: A* executed on board-2-1. Dots represents the best path A* calculated, x represents a closed node and * represents an open node, or a node in the frontier

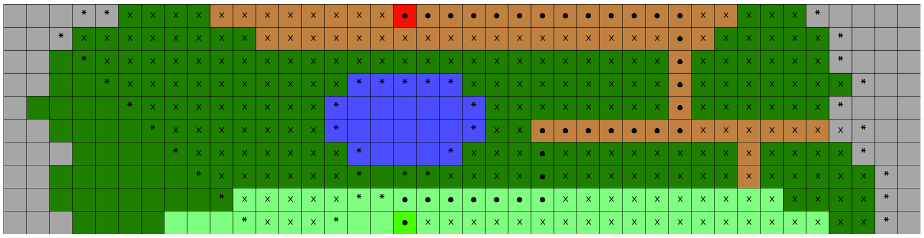


Figure 3.14: Dijkstra executed on board-2-1. Dots represents the best path dijkstra calculated, x represents a closed node and * represents an open node, or a node in the frontier

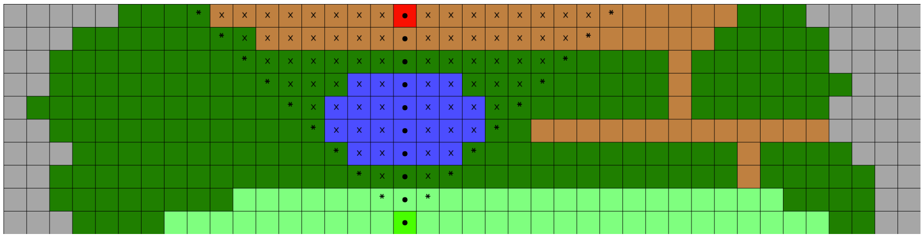


Figure 3.15: Breadth first search executed on board-2-1. Dots represents the best path breadth first search calculated, x represents a closed node and * represents an open node, or a node in the frontier

Comparison of Board 2-1 solutions: Here it gets interesting. In this case, bfs does not find an optimal path: the one leading directly over the pond of water is slow, but the first to be found, and thus accepted. The difference between dijkstra and A* is more subtle. A* takes a step forwards towards the goal before moving towards the left. The cost of the paths is the same, and dijkstra does not tell the difference, but from the informed point of view, moving forward is better because it brings the agent closer to the goal, and thus a lower heuristic cost, than moving to the left. Even if both algorithms found an optimal path, A* did so expanding 30 fewer nodes (237 vs 267, see stat.txt).

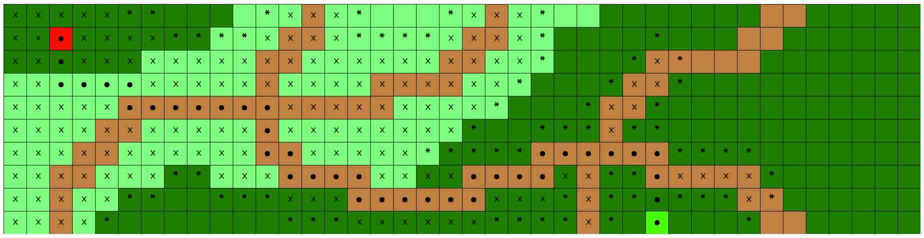


Figure 3.16: A* executed on board-2-2. Dots represents the best path A* calculated, x represents a closed node and * represents an open node, or a node in the frontier

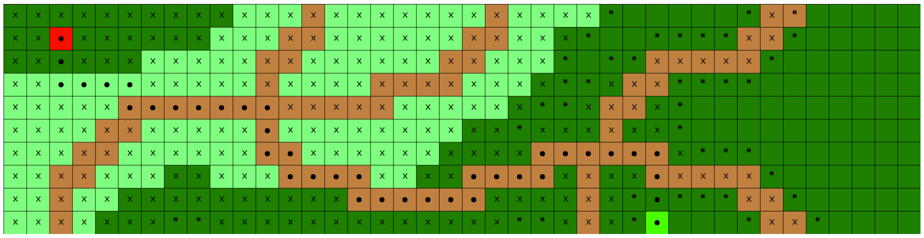


Figure 3.17: Dijkstra executed on board-2-2. Dots represents the best path dijkstra calculated, x represents a closed node and * represents an open node, or a node in the frontier

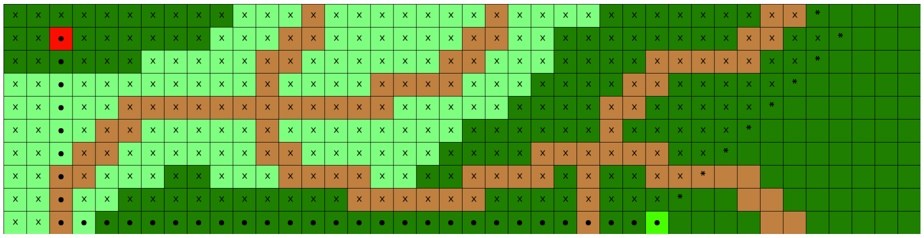


Figure 3.18: Breadth first search executed on board-2-2. Dots represents the best path breadth first search calculated, x represents a closed node and * represents an open node, or a node in the frontier

Comparison of Board 2-2 solutions: The difference between dijkstra and A* is even clearer is this example. Djikstra follows the roads because each step is cheap, even if it leads away from the goal. This leads it to make an expensive journey around the board, while A* follows the road leading towards the goal.

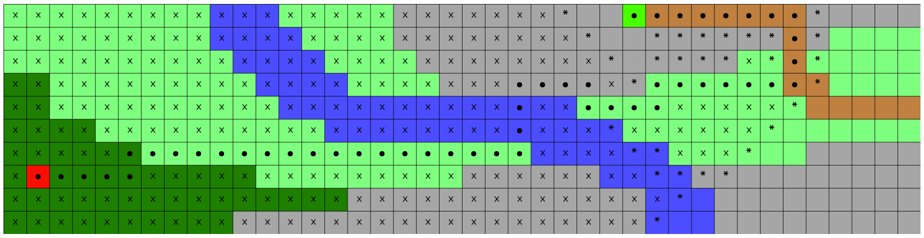


Figure 3.19: A* executed on board-2-3. Dots represents the best path A* calculated, x represents a closed node and * represents an open node, or a node in the frontier

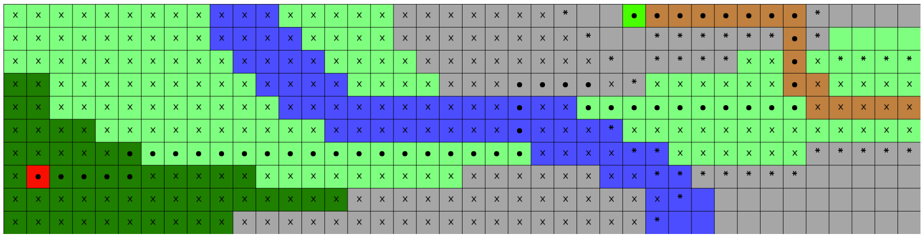


Figure 3.20: Dijkstra executed on board-2-3. Dots represents the best path dijkstra calculated, x represents a closed node and * represents an open node, or a node in the frontier

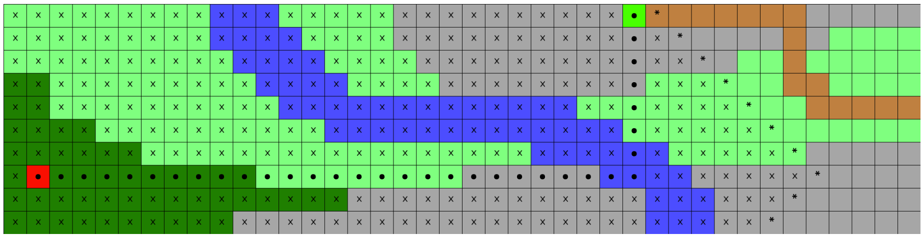


Figure 3.21: Breadth first search executed on board-2-3. Dots represents the best path breadth first search calculated, x represents a closed node and * represents an open node, or a node in the frontier

Comparison of Board 2-3 solutions: In this case we seem the same greedy road-following behavior from dijkstra.

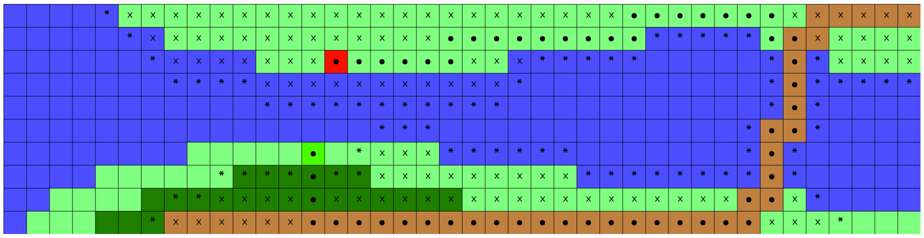


Figure 3.22: A* executed on board-2-4. Dots represents the best path A* calculated, x represents a closed node and * represents an open node, or a node in the frontier

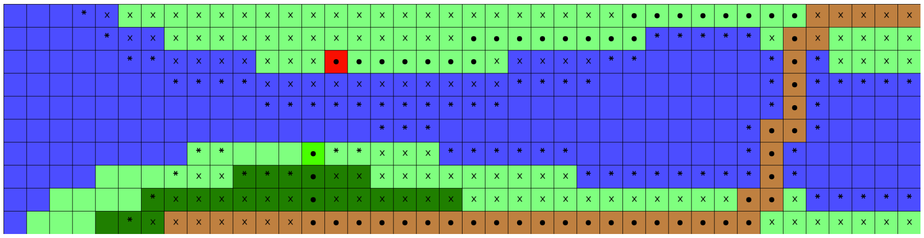


Figure 3.23: Dijkstra executed on board-2-4. Dots represents the best path dijkstra calculated, x represents a closed node and * represents an open node, or a node in the frontier

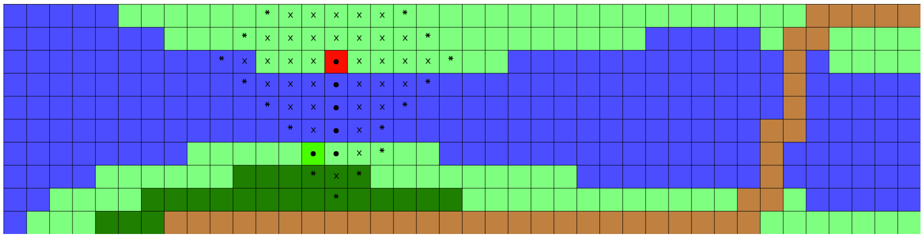


Figure 3.24: Breadth first search executed on board-2-4. Dots represents the best path breadth first search calculated, x represents a closed node and * represents an open node, or a node in the frontier

Comparison of Board 2-4 solutions: BFS again exhibits its inefficient love of swimming. A* and dijkstra is again very similar, because they both tend towards the cheap road.