



COMP 472 MP 2

Analysis

Robin Laliberté-Beaupré - 40180181
Sean McKenzie - 40068618
Hristo Marinov - 40156820



Length of the solutions across algorithms and heuristics

UCS

- Uniform Cost Search algorithm always finds the lowest-cost solution
 - Whenever we expand a node, we insert the new nodes in the open list sorted by the cost to reach them from the root
 - This guarantees that whenever the algorithm finds a solution, this solution has the lowest possible cost.
- In our results, the lowest-cost solutions are always found by UCS



Length of the solutions across algorithms and heuristics

GBFS


- GBFS does not guarantee to find lowest-cost solutions
 - Greedy Best First Search algorithm does not take into account the cost to reach a node from the root when it is added to the open list
 - Instead, newly expanded nodes are sorted only according to their heuristic value
- In our results, GBFS often does not find the lowest-cost solution
 - This is especially true for the more complex puzzles with longer solution paths. In those cases, GBFS generally finds significantly longer solutions than UCS and A*.
- The first three heuristics perform the same for GBFS
 - H1 and H2 give the same values for all our sample boards
 - H3 has the same impact on ordering in the open list as H1 in GBFS
 - H4 gives a few different length solutions for GBFS, both shorter and longer



Length of the solutions across algorithms and heuristics

A/A*

- H1 and H4 always give lowest-cost solutions with algorithm A* since they are admissible
- H2 also performs as an admissible heuristic in our tests and gives lowest-cost solutions
- These results are as expected since algorithm A/A* considers the cost to reach a node from the root when sorting the open list
- Since H3 is not admissible, it does not give lowest-cost solutions with algorithm A
 - This is as expected since using a non-admissible heuristic means the cost of some nodes is overestimated when they are inserted in the open list
 - Thus, the algorithm cannot reliably eliminate costlier solution paths
- In our results, algorithm A with H3 with a constant of 3 gives solution paths that are in-between the high-cost paths of GBFS and the lowest-cost solution



Admissibility of each heuristic and its influence on the optimality of the solution

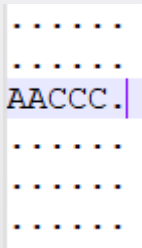
H1: The number of blocking vehicles

- H1 is admissible
 - A minimal number of moves for a solution to a puzzle is one move per car blocking the exit plus one move to bring the ambulance to the exit
 - Since H1 measures the number of cars blocking the exit, it will at most be one less than the actual cost of the solution
 - In practice H1 is usually much lower than the actual cost
- Since H1 is admissible, the solutions found by the A/A* algorithm will always be optimal
- This is true in our results


Admissibility of each heuristic and its influence on the optimality of the solution

H2: The number of blocked positions

- H2 is not admissible. To show this, consider the following board:




- The solution to this board is C -> 1, A -> 4 which has a cost of 2
- H2 would give a value of 3 since there are three blocked squares between A and the exit
- This can only occur in the unusual configuration depicted above of having a car of length 3 in the horizontal direction on the third rank



Admissibility of each heuristic and its influence on the optimality of the solution

H2: The number of blocked positions

- In practice, H2 usually performs as an admissible heuristic
- This was the case in all our sample inputs
- For this reason, the inadmissibility of H2 did not have an impact on the optimality of the solutions found
- All solutions found by algorithm A/A* using H2 were optimal



Admissibility of each heuristic and its influence on the optimality of the solution

H3: The value of H1 multiplied by a constant λ of your choice, where $\lambda > 1$

- H3 is not admissible
 - A minimal number of moves for a solution to a puzzle is one move per car blocking the exit plus one move to bring the ambulance to the exit
 - If we multiply the value of the number of blocking cars by any constant larger than 1, our estimate automatically becomes larger than the minimal cost solution
 - Thus, H3 usually overestimates the cost of the solution
- Many solutions found by algorithm A/A* with H3 are non-optimal
- It does not, however, have an impact on the length of the solutions found by GBFS

Admissibility of each heuristic and its influence on the optimality of the solution

H4: The number of blocking vehicles + the largest number of vehicles blocking the blocking vehicles blocked on both sides

Explanation

- The first part of the heuristic is H1
- The second part of the heuristic looks at each blocking vehicle and compares the number of vehicles that block those on either side:

1. If either side (up or down) of the vehicle is free (0 blocking vehicles), then this

.....B.
AA...B.
.....CC
.....DD
.....EE

$H4 = H1 + 0 = 1$

Actual cost of
solution: 2

Admissibility of each heuristic and its influence on the optimality of the solution

H4: The number of blocking vehicles + the largest number of vehicles blocking the blocking vehicles blocked on both sides

2. If both sides of the vehicle (up and down) have at least one blocking vehicle, then we take the smallest number of them

```
....FF
....B.
AA..B.
....CC
....DD
....EE
```

$$H4 = H1 + 1 = 2$$

Actual cost of
solution: 3

- For each blocking vehicle blocked on both sides we take the smallest number of blocking vehicles on either side and then take the largest of all of those
- This is because at minimum we need to free the blocking car that is blocked by the largest number of cars

Admissibility of each heuristic and its influence on the optimality of the solution

H4: The number of blocking vehicles + the largest number of vehicles blocking the blocking vehicles blocked on both sides


- This can take either one move per blocking car or one move per blocking car minus one

```
...FFF
....GG
AAHBC.
..HBC.
....DD
...EEE
```

$$H4 = H1 + 2 = 5$$

Actual cost of
solution: 5

- Car C is blocked on both sides by 2 cars. Car B is blocked by 1 car on both sides and car H is blocked by none. The largest of those is 2.
- Note: freeing the blocking car blocked by the largest number of cars can take one move per blocking car *minus one*. This does not affect the admissibility since H4 never counts the move of the ambulance to the exit.




Admissibility of each heuristic and its influence on the optimality of the solution

H4: The number of blocking vehicles + the largest number of vehicles blocking the blocking vehicles blocked on both sides

Admissibility

- H4 is admissible
 - H1 is admissible
 - We add to H1 either:
 - The minimum cost to free all the cars blocking A, or
 - The minimum cost to free all the cars blocking A plus one
 - A minimal cost of the solution is:
 - The minimum cost of freeing all the cars blocking A + One move per car blocking A (H1) + One move to bring A to the exit
 - H4 never includes the cost of bringing A to the exit
 - Thus, H4 is at most equal to the minimal cost of the solution



Admissibility of each heuristic and its influence on the optimality of the solution

H4: The number of blocking vehicles + the largest number of vehicles blocking the blocking vehicles blocked on both sides

- In practice, H4 is usually much smaller than the actual cost of the solution
- The admissibility of H4 means it finds optimal solutions to all puzzles using algorithm A/A*
- This is shown by our testing, in which all puzzles using A/A* find optimal solutions



The execution time across algorithms and heuristics

UCS

- In our results Uniform Cost Search was generally slower than informed search
- It is faster for boards with no solutions since it searches as many states as informed search but without having to compute heuristics

GBFS

- GBFS is always the fastest algorithm for every input board with a solution
- This is an expected result given that GBFS will return the first solution it finds no matter its actual cost
- The first three heuristics give similar search paths and runtimes for GBFS
- H4 gives shorter search paths and runtimes in most tests
 - Since GBFS aims to minimize the value of the heuristic as quickly as possible, having a more accurate value allows it to more quickly reach a solution.



The execution time across algorithms and heuristics

A/A*


- The A/A* algorithm has varying search path lengths and runtimes depending on which heuristic is used
- For all heuristics, the cost to compute heuristic values and check the open/closed list for repeated nodes starts to take a toll for more complex puzzles
- H1 and H2 perform the same for all tests
 - The search paths they give are always longer than that of GBFS and UCS
 - Because of the added costs, the runtimes of algorithm A* with H1 and H2 are longer or equal for complex puzzles than the runtime of UCS



The execution time across algorithms and heuristics

A/A*

- Algorithm A performs very close to GBFS when using H3 with a constant value of 3
 - This can be explained by the constant multiplier inflating the heuristic value of nodes
 - Because of this, the ordering in the open list is based much more on heuristic values than on the cost to reach nodes, so it resembles the ordering in GBFS
 - Thus, algorithm A with H3 finds solutions much more quickly than with the other heuristics
 - These solutions are not always optimal, but they are generally better than the ones found by GBFS
 - However, the cost to check the open/closed list for repeated nodes means that runtimes of this algorithm are longer than that of GBFS
 - Still, algorithm A with H3 is an interesting option for quickly finding low-cost solutions



The execution time across algorithms and heuristics

A/A*

- H4 performs better than H1 and H2 in terms of search paths lengths and runtime
- The improvement in runtime, however, is not less than would be expected given the shorter search paths
 - This is an expected result as it takes longer to compute H4 than H1 and H2
- As with the other three heuristics, the cost to compute heuristics and check the open/closed list for repeated nodes starts to take a toll for more complex puzzles
- Thus, algorithm A* with H4 has a runtime close to that of UCS for puzzles with longer search/solution paths
- It is always much slower than GBFS.



The execution time across algorithms and heuristics

Summary

- If the goal is to find solutions as quickly as possible, with no regard for their cost, algorithm GBFS is the obvious choice
 - Algorithm A with heuristic H3 is an interesting alternative which finds lower-cost solutions, albeit through longer runtimes than GBFS
- If optimal solutions need to be found, the best choice overall is algorithm A* with H4
 - This is especially true for simpler puzzles
- A* with H1 and H2, on the other hand, does not provide a reliable improvement in runtime over UCS
- In our tests, informed search with an accurate heuristic is almost always faster than uninformed search to find lowest-cost solutions. However, informed search with an inaccurate heuristic may not be.
- Improvements to our choices of data structures and searching/sorting strategies could potentially have significant impacts on the runtimes of the different algorithms
- However, the results we obtained are broadly compatible with the theory