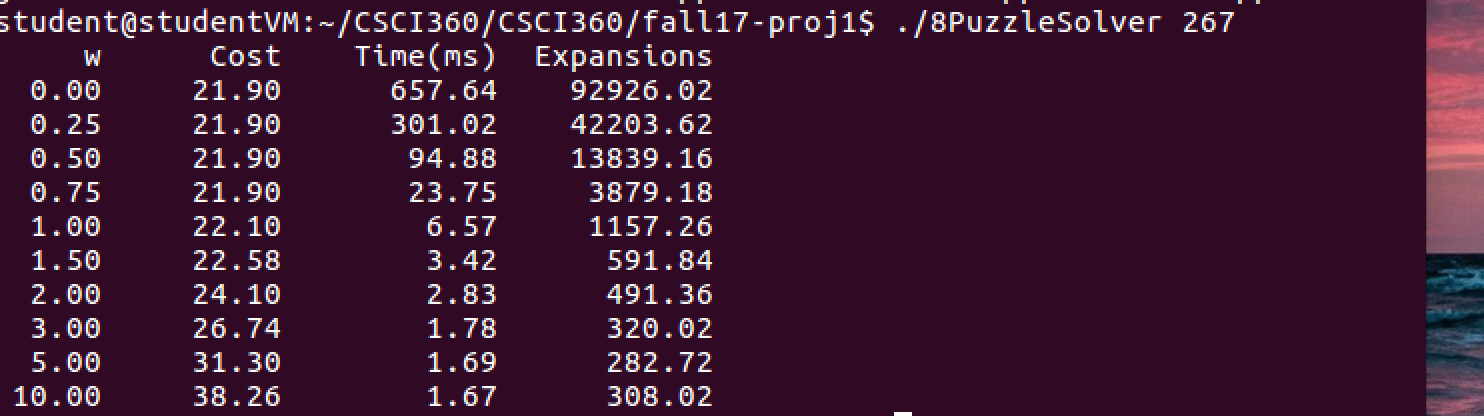
Aaron Schrock

2511656267

09/25/2017

Project 1: Theoretical Part



1. What trends do you observe in runtime, number of expansions, and the resulting

path length as w varies?

As W increases the runtime gets shorter, the number of expansions decreases, and the resulting path length increases.

2. For which values of w is Weighted A\* guaranteed to find a shortest path (and

why)? For which values of w is Weighted A\* not guaranteed to find a shortest

path (and why)?

For w <= 1.0 A\* is guaranteed to find the shortest path because the heuristic is still

admissible. f(n) = 1\*h(n) + g(n) is admissible when h(n) = Manhattan distance, and if you

decrease the w value below 1, f(n) will only get smaller, which will still be an underestimate

of true cost. Therefore for 1, and all values less than 1, f(n) is admissible and by definition

guaranteed to find the shortest path.

For all values of w greater than 1, A\* is not guaranteed to find the shortest path because the

heuristic may no longer be admissible. As described earlier, f(n) = 1\*h(n) + g(n) is admissible using Manhattan distance, but if you increase the w value to be greater than 1, f(n) may no longer be an underestimate of the true cost.

For example: (1\*h(n) + g(n)) <= c(n) where c(n) is the true cost is valid.

But for some inputs, (5\*h(n) + g(n)) is not necessarily <= c(n) and could be greater because of the higher multiple, making f(n) not admissible, and therefore A\* is not guaranteed to find the shortest path.

3. What can you say about the solution cost as w increases? Why do you think

that this is the case?

The solution cost increases as w increases. This is the case because once w is greater than 1, the heuristic is no longer admissible, and A\* is not guaranteed to find the shortest path. As w continues to increase above 1, the heuristic becomes a larger underestimate, which can lead to a non-optimal decision in which move to choose next, causing a higher cost over all. The algorithm may end up doing superfluous work examining paths that should be ignored and potentially finding non-optimal paths with a higher cost.

4. What can you say about the number of expansions as w increases? Why do you

think that this is the case?

The number of expansions decreases as w increases because g(n) makes less of an impact on f(n) the larger that w is. if w is 10, h(n) will be so large, that it will make g(n) nominal when compared to smaller w values. This leads to the algorithm going deeper into paths, even if they may not be optimal because anyway the path may be several steps in, the decrease in h(n) as it approaches the goal state will still be cheaper than starting a new path with an amplified h(n) value and small g(n) value. Many of these non-optimal paths will still lead to a solution just in many steps (high cost). Eventually the algorithm will reach the goal state going down one of these non-optimal paths and will return that the solution is found. By going deeper down each individual path, rather than backtracking and trying new potentially more optimal paths the solution will be reached sooner. Because of this, solutions will be found sooner and with less states expanded.

5. What can you say about the runtime as w increases? Why do you think that

this is the case?

The runtime decreases as w increases because less expansions are performed as per question 4. The time saved from exploring fewer expansions leads to this decreased run times.

6. Suppose that we modify the Manhattan distance heuristic so that it also treats

the blank position as a tile (that is, the summation also includes the distance of

the blank position to its goal position). What are the advantages and disadvantages

of using the modified heuristic in conjunction with a regular (unweighted)

A\* search? Is the modified heuristic admissible?

The heuristic is no longer admissible (it can potentially become an overestimate), which is a disadvantage because the algorithm is not guaranteed to find the most optimal solution.