Homework #1

Wenzheng Zhang, Qi Wu

February 28, 2020

Part 1

a

The estimated cost of cheapest solution through the east neighbor of the agent is

$$f(east) = g(east) + h(east) = 1 + h(east) = 1 + 2 = 3$$

But the estimated cost of cheapest solution through the north neighbor of the agent is

$$f(north) = g(north) + h(north) = 1 + h(north) = 1 + 4 = 5$$

According to the rule of A* search, because $f(east) < f(north)$, we choose to move the agent to the east at the first step given that the agent doesn't know initially which cells are blocked.

b

We keep a closed list and the agent never comes back to the cells in the closed list. Because the grid world is finite, we can put at most finite number of cells into the closed list, which means we can only move finite steps to find the target or report that it is impossible to reach the target if we cannot find unblocked neighbors at one cell.
Suppose there are n unblocked cells in the grid world. For each iteration, the agent can move at most n cell and find at least one blocked cell neighboring one unblocked cell. Since there are at most 4n blocked cells neighboring unblocked cell, the algorithm can only run at most 4n times. Therefore, that the number of moves of the agent until it reaches the target or discovers that this is impossible is $O(n \times 4n) = O(n^2)$.

Part 2

We set 2 different modes of priority queue for the open list in A* search.For both modes, we first take f-values as priorities in favor of smaller f-value. When we encounter two cells with the same f-values, if we set mode=0, we use $g(s)$ as priorities to break ties in favor of cells with smaller g-values. If we set mode=1, we use $-g(s)$ as priorities to break ties in favor of cells with larger g-values. For the same maze we generate, we run A* search using two different modes of priority queue. Figure 1 is the results of 50 different experiment:

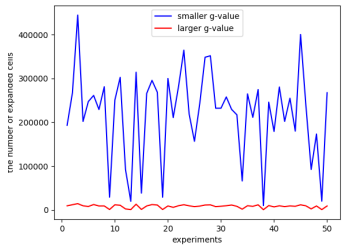
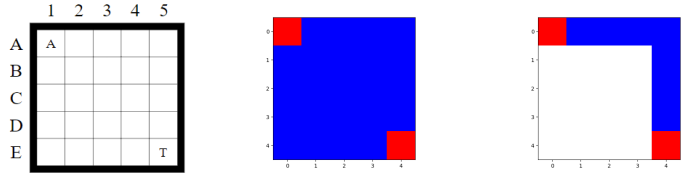


Figure 1: Smaller cost and larger cost

The y-axis is the total expanded cells that the agent starts from the start until it reaches the target or finds no way to the target in each experiment, the x-axis is each experiment. From the Figure we can know the number of expanded cells of mode=0 is always bigger than the number of expanded cells of mode=1, which means the Repeated Forward A* with larger g-value tie break strategy is much faster than Repeated Forward A* with smaller g-value tie break strategy.
The reason is that the agent will choose the point closer to the target to expand if we choose larger g-value to break ties but the agent will choose the point closer to the start to expand if we choose smaller g-value to expand and we will expand many useless cells in this case. Take two states s_1, s_2 with the same f-value $f = f(s_1) = f(s_2)$ but $g(s_1) < g(s_2)$ as an example. We know $f(s_1) = g(s_1) + h(s_1)$ and $f(s_2) = g(s_2) + h(s_2)$, so $h(s_1) > h(s_2)$. Because $f(s_1) = f(s_2)$ and for mode 0 we choose smaller g-value to break tie, which means we will choose s_1 to expand since $g(s_1) < g(s_2)$. Thus, we choose the state with larger h-value to expand, which means we choose to expand the state that is closer to the start states and expand all other states with the same f-value and g-value as s_1 before we can expand the states with the same f-value but with smaller h-value in this case. Similarly,

for mode 1 we choose larger g-value to break tie, which means we choose s_2 to expand. Because $h(s_1) > h(s_2)$, we choose the point that is closer to the target in this case and don't need to expand any other states that have the same f-value and g-value with s_2 before we expand the state with the same f-value but smaller h-value. Let's consider the following example to see which cells the two modes will choose to expand.



(a) maze (b) smaller (c) larger

Figure 2: result of larger cost and smaller cost. the blue cells is the expanded cells

For mode 0, see Figure 2(c). The cells in blue means that the cells have been expanded through the whole search process.
For mode 1, see Figure 2(b).
From these two results, we can see that we expand the cells closer and closer to the target when we choose larger g-value but we expand all the cells when we choose smaller larger g-value. For mode 1(larger g-value to break tie), here is the order of the expansion of the cells

$$A_1 \to A_2 \to A_3 \to A_4 \to A_5 \to B_5 \to C_5 \to D_5 \to E_5$$

For mode 0(smaller g-value to break tie), here is the order of expansion (we always choose the point that is put into the open list earlier to expand when both f-value and g-value are equal).

$$\begin{aligned} & A_1 \\ & \to A_2 \to B_1 \\ & \to A_3 \to B_2 \to C_1 \\ & \to A_4 \to B_3 \to C_2 \to D_1 \\ & \to A_5 \to B_4 \to C_3 \to D_2 \to E_1 \\ & \to B_5 \to C_4 \to D_3 \to E_2 \\ & \to C_5 \to D_4 \to E_3 \end{aligned}$$

$$\rightarrow D_5 \rightarrow E_4$$

$$\rightarrow E_5$$

We can see that we expand cells level by level ordering by the distance to the start point and expand many useless points when we choose smaller g-value to break tie.

part 3

The Repeated Forward A* and Repeated Backward A* can share same A* searching algorithm. Therefore, to implement these 2 algorithm, we add a parameter $decode_mode$ to our A* function. When $decode_mode = 1$, the function will swap the position of *start* and *goal* and invert the *path* to do Backward A* searching.

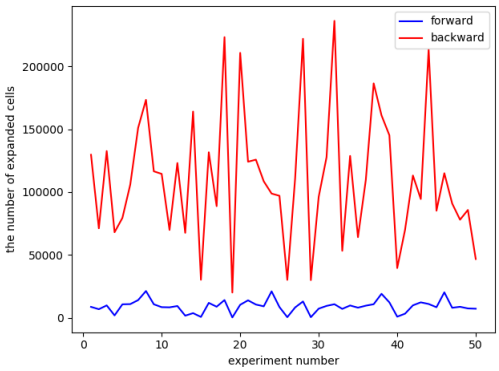


Figure 3: Forward and Backward

Figure 3 show the result of 50 test on 101×101 maze. According to the result, Repeated Forward A* expanded much less nodes than Repeated Backward A*. On average, Repeated Forward A* only expand 9113 nodes in each test, whereas Repeated Backward A* expand 111250 nodes. In addition, the variance of Repeated Forward A* is 24296811, and the variance of Repeated Backward A* is 2758697079. So, Repeated Forward A* is more stable than Repeated Backward A*.
We believe the reason is that the known world is closer to *start* than *goal*. In general, for any two points, if there are more blocked cells around one point, the cost of the path between them will be expectedly higher. Since the known world is closer to *start*, the blocked cells are relatively close to *start*.

Therefore, for any two cells a and b respectively in forward and backward search, if $h(a) = h(b)$, there are higher probability that $g(a) > g(b)$. So, in general, there are more cells s with $f(s) < gd(start)$ in backward a* search than in forward a* search, and Backward A* search will expand more cells.

Part 4

Proving Manhattan distances are consistent:

According to triangle inequality, for any three point a, b and c, we have

$$h(a, c) \leq h(a, b) + h(b, c)$$

$h(a, c)$ is the Manhattan distance between a and c.
Then, since the agent can move only in the four main compass directions, the shortest distant between any two points is their Manhattan distances. So,

$$h(a, b) + h(b, c) \leq c(a, b) + h(b, c)$$

$c(a, b)$ is the real distance between a and b. So, any two point a and b satisfy

$$h(a, c) \leq c(a, b) + h(b, c)$$

Therefore, Manhattan distance are consistent.

Proving Adaptive A* is constances

Proving by induction.
Since the initially h-values are consistent, we assume the h-values in first $i-1$ searching are consistent, and need to prove the h-values in i^{th} searching are consistent. Separating the situation of any two cell a and b into three cases and prove

$$h^i(a) \leq c^i(a, b) + h^i(b)$$

1. Both a and b were expanded in $i - 1^{th}$ search.
   Because both a and b were expanded, their h-values in i^{th} search are

   $$h^i(a) = g^{i-1}(goal) - g^{i-1}(a) \text{ and } h^i(b) = g^{i-1}(goal) - g^{i-1}(b)$$

   According to triangle inequality, we know

   $$g^{i-1}(b) \leq g^{i-1}(a) + c^{i-1}(a, b)$$

Therefore, because the action cost is nondecreasing

$$\begin{aligned} h^i(a) &= g^{i-1}(goal) - g^{i-1}(a) \\ &\leq g^{i-1}(goal) - g^{i-1}(a) + c^i(a,b) \\ &\leq h^i(b) + c^i(a,b) \end{aligned}$$

2. a was expanded but b was not.
   Because a was expanded but b wasn't, their h-values in i^{th} search are

   $$h^i(a) = g^{i-1}(goal) - g^{i-1}(a) \text{ and } h^i(b) = h^{i-1}(b)$$

   According to triangle inequality, we know

   $$g^{i-1}(b) \leq g^{i-1}(a) + c^{i-1}(a,b)$$

   Because b wasn't expanded, according to the expanding rule of A*, we have

   $$g^{i-1}(goal) \leq g^{i-1}(b) + h^{i-1}(b)$$

   Therefore,

   $$\begin{aligned} h^i(a) &= g^{i-1}(goal) - g^{i-1}(a) \\ &\leq g^{i-1}(b) + h^{i-1}(b) - g^{i-1}(a) \\ &\leq h^i(b) + c^i(a,b) \end{aligned}$$

3. b was expanded but a wasn't expanded.
   In this case,

   $$h^i(a) = h^{i-1}(a) \text{ and } h^i(b) = g^{i-1}(goal) - g^{i-1}(b)$$

   Because the heuristic action cost keeps nondecreasing, we can know

   $$h^i(a) = h^{i-1}(s) \leq h^{i-1}(b) + c^{i-1}(a,b) \leq h^i(b) + c^i(a,b)$$

Therefore, we prove that Adaptive A* leaves initially consistent h-values consistent

Part 5

Figure 4 shows result of 50 independent tests for repeated A* search and adaptive A*.
From this result figure we can know the total expanded states of these two different search algorithm is nearly equal but adaptive A* search expands a little less states than repeated A* search. I think the reason is that adaptive A* search is more informed, that is, the heuristic of adaptive A* search is closer to the real heuristic.