Setup Your Environments

The maze is generated by a DFS through the maze and traversing each cells neighbor randomly. Before every cell is entered into the open list, we generate a random integer from 1-100. If the integer is greater than 70, the cell is marked as block by making its cost infinity, otherwise the cost is kept as 1.

Understanding the Methods

A* algorithms expand nodes with the lowest f-values first. Based on this expansion, whichever path reaches the goal first, is returned by the algorithm. F-values are the sum of g-values (cost from start to current state) and h-values (in this case, Manhattan Distances). In this example, A (E2) would be popped out of the open list and we traverse its neighbors E1, E3, and D2. They are inserted into the open list with their respective g-values, h-values, and f-values:

Cell	Parent	G-value	H-value	F-value
E3	E2	1	2	3
D2	E2	1	4	5
E1	E2	1	4	5

Since E3 has the lowest f-value, it will be expanded first, resulting in the following open list:

Cell	Parent	G-value	H-value	F-value
E4	E3	2	1	3
D3	E3	2	3	5
D2	E2	1	4	5

E1	E2	1	4	5

Note we do not insert previously expanded nodes into the open list. E4 will be popped out next and expanded to give the following open list:

Cell	**Parent**	**G-value**	**H-value**	**F-value**
E5	E4	3	0	3
D4	E4	3	2	5
D3	E3	2	3	5
D2	E2	1	4	5
E1	E2	1	4	5

At this point the goal, E5, will be popped out, and the path would be determined as: E2 (A), E3, E4, E5. The path returned by A* will, therefore, move east first because these cells have smaller f-values.

There are two main reasons why exploring the gridworld takes a finite amount of time:

The gridworld has a finite number of cells

Each cell is only expanded once in an A* search to avoid falling into infinite loops.

The agent traverses neighbors in search of the goal and adds these cells to the open list. If the goal is popped from the open list, the search successfully ends and returns a path from the start to the goal. Otherwise, the search would end when each cell that is reachable from the start has been expanded and there are no more cells to expand. This means that the goal has not been popped, and therefore cannot be reached from the start. Since the gridworld finite, and each cell is on expanded once, we would find a path in a finite amount of time.

To find the upper bound number of moves for Repeated A* search, let us use the formula:

$$\#\,moves \;=\; \#\,of\ A^{*}\ plannings \;*\; \frac{\#moves}{planning}$$

In the case where the goal is reachable from the start:

The worst case would occur when all the unblocked cells are reachable from the start. For each planning, the same cells cannot be expanded twice, therefore cannot be included in the planned path more than once. Hence, the maximum number of moves would require the agent to step through all unblocked cells once to reach the goal, so assuming $n \;=\; number\ of\ unblocked\ cells$, $\frac{\#moves}{planning} \;\leq n$.

The number of A* plannings depends how far the agent traveled in each A* execution before reaching a blocked cell. Additionally, with each new start the agent has knowledge of it's immediate blocked neighbors. With these two facts in mind, in the worst case, the agent makes only one successful move in the path execution before discovering a blocked cell. This would result in a new A* planning with a new start. In the worst case, each planning would start with the very next unblocked cell, therefore iterating through each unblocked cell (n) to reach the goal, so $\#\,of\ A^{*}\ plannings \;\leq n$.

To get the upper bound number of moves for Repeated A*, we multiply the maximum number of A* plannings by the number of moves per planning:

$$\#\,moves \;=\; \#\,of\ A^{*}\ plannings \;*\; \frac{\#moves}{planning}$$

$$\#\,moves \;\leq\; n \;*\; n$$

$$\#\,moves \;\leq\; n^{2}$$

In the case where the goal is not reachable from the start:

The worst case would occur under the same conditions as the last case, however instead of no unreachable unblocked cells, there exists only one unblocked cell unreachable from the start, which is the goal cell. With everything else the same, the maximum number of moves per planning is still n. The maximum number of A* plannings is still n, therefore the upper bound moves is also n^{2}

Heuristics in the Adaptive A*

Manhattan distances are consistent if the following inequality holds true for the heuristic:

$h(n) \;\leq h(s) \;+ C(n,s),\ where\ n\ and\ s\ are\ states$	
$h(n) \;=\; \lvert x_n - x_{goal}\rvert + \lvert y_n - y_{goal}\rvert$, $h(s) \;=\; \lvert x_s - x_{goal}\rvert + \lvert y_s - y_{goal}\rvert$.	
$h(n) \;- h(s) \;\leq C(n,s)$	

$\lvert x_n - x_{goal}\rvert + \lvert y_n - y_{goal}\rvert - \lvert x_s - x_{goal}\rvert - \lvert y_s - y_{goal}\rvert \leq C(n,s)$	Substitution
$\lvert x_n - x_s\rvert + \lvert y_n - y_s\rvert \leq C(n,s)$	Simplify
$h(n,s) \leq C(n,s)$, $h(n,s)$ = Manhattan Distance from state n to s	
Manhattan takes the sum of horizontal and vertical distance between two states. Because the agent only travels in the 4 cardinal directions, Manhattan distance represents the minimum possible cost of travelling between two cells. This occurs when the path is completely unblocked. From $h(n,s)$, the cost can only increase to circumvent blocked cells. Manhattan distance would not be consistent if the agent was allowed to move in diagonals, as the $C(n,s)$ can be shorter if the path was unblocked there.	
If the agent is allowed to move in diagonals:	
$C(n,s) = \sqrt{(x_n - x_s)^2 + (y_n - y_s)^2}$, *on an unblocked path*	
Assume $x_n - x_s = a,\ y_n - y_s = b,\ \lvert a\rvert + \lvert b\rvert \leq \sqrt{a^2 + b^2}$	
$\lvert a\rvert + \lvert b\rvert \leq \sqrt{a^2 + b^2}$	Does not abide by the triangle inequality.

h_{new} are also consistent. To prove this we have 3 cases:

1. Neither n nor s have been previously expanded (proved above)
2. States n and s have both been expanded previously
3. Only state n has been previously expanded (similar proof for if s has not been previously expanded)

Case 2: States n and s have both been expanded previously. Subscript 1 represents values from previous searches, whereas 2 represents current search.

$h_{new}(n) \leq h_{new}(s) + C_2(n,s)$, *where n and s are states*	
$h_{new}(n) = C_1(start,\ goal) - g_1(n)$ $h_{new}(s) = C_1(start,\ goal) - g_1(s)$	
$h_{new}(n) - h_{new}(s) \leq C_2(n,s)$	
$C_1(start,\ goal) - g_1(n) - C_1(start,\ goal) + g_1(s) \leq C_2(n,s)$	Substitution

$g_1(s) - g_1(n) \leq C_2(n,s)$	Simplify
$C_1(n,s) \leq C_2(n,s)$	Substitute
With each successive planning of A*, the cost between cells must increase or remain the same because when new blocked cells are discovered, they increase the path cost, and therefore, the distance between some cells. The path cost from one planning to the next within Repeated A* path finding does not decrease. The difference between $g_1(s)$ and $g_1(n)$ is the path cost of n and s from the previous search. Since the cost is nondecreasing for each successive search, the inequality holds true.	

Case 3: Only state n has been expanded in a previous A* search (similar proof for if *s* has not been previously expanded)

$h_{new}(n) \leq h(s) + C_2(n,s)$, *where n and s are states*	
$h_{new}(n) = C_1(start, goal) - g_1(n)$ $h(s) = \lvert x_s - x_{goal}\rvert - \lvert y_s - y_{goal}\rvert$	
$h_{new}(n) - h(s) \leq C_2(n,s)$	
$C_1(start, goal) - g_1(n) - \lvert x_s - x_{goal}\rvert + \lvert y_s - y_{goal}\rvert \leq C_2(n,s)$	Substitution
$C_1(n, goal) - \lvert x_s - x_{goal}\rvert + \lvert y_s - y_{goal}\rvert \leq C_2(n,s)$	Simplify
This case assumes the most optimistic path between *s* to *goal*, meaning that the node *s* has not been considered before, therefore must be a node A* is using to circumvent newly discovered blocked cells. Assuming the distance from *s* to *goal* is in fact a direct path, then the cost of the current search, $C_2(n,s)$, will result in a minimum distance between *n* and *s* calculated by the left side of the inequality. $C_2(n,s)$ cannot be lower than this distance, as this distance assumes there are only as many blocks between *n* to *s* as previously discovered in the path, however there may be more blocks resulting in a higher $C_2(n,s)$.	

Heuristics in the Adaptive A*

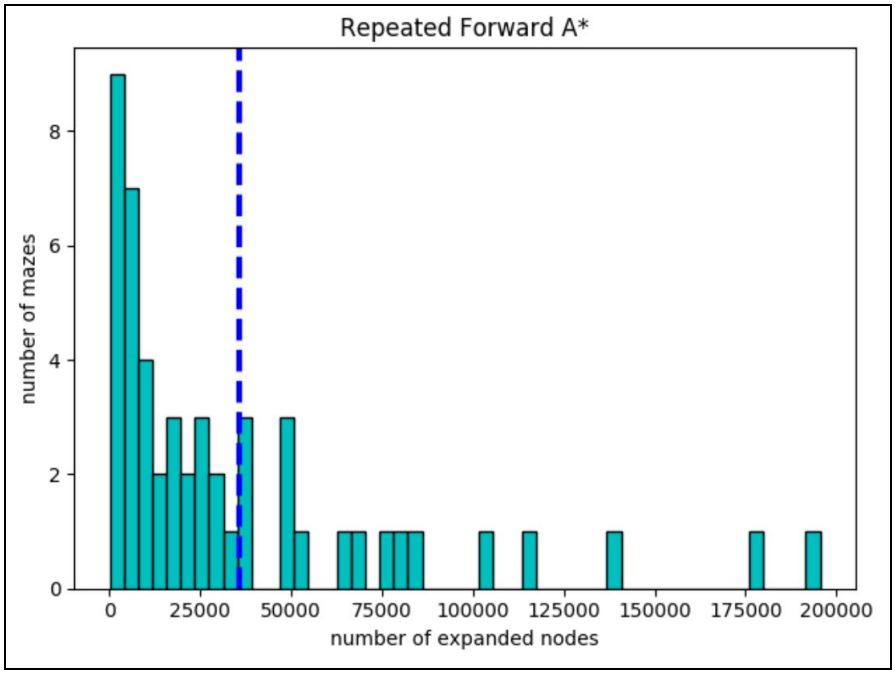

Figure 5

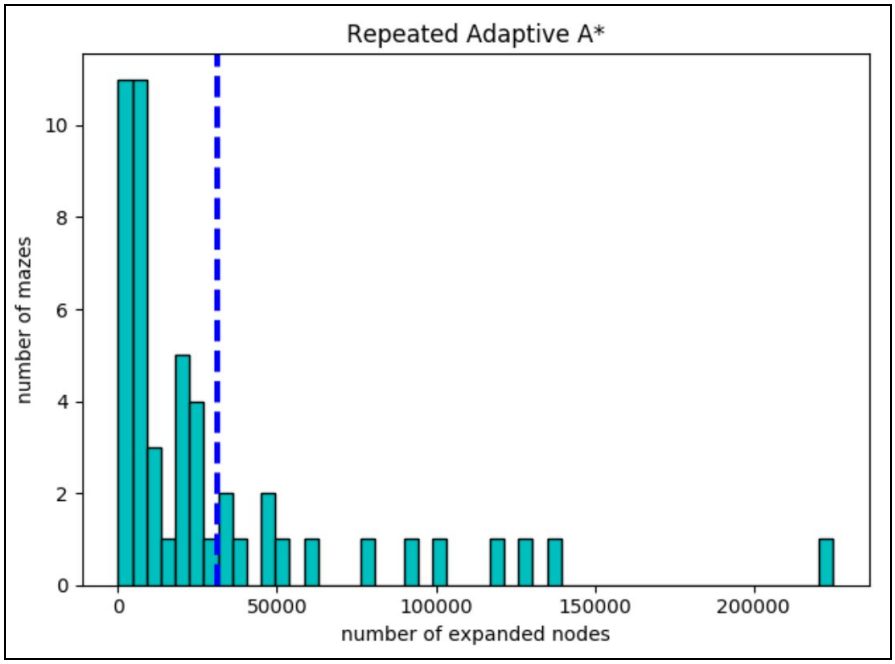

Figure 6

We measured the amount of expanded cells to compare forward A* and adaptive A*. The average number of expanded cells for forward A* is 35,733 whereas the average for adaptive A*

31, 296. The number of cells expanded by adaptive A* is therefore less than the number of cells expanded by forward A*. This is because adaptive A* is a more focused search as it takes in account more accurate heuristics with each successive search. With adaptive A*, the cost of the actual path due to blocked cells is taken into account, therefore cells previously assumed to be closer to the goal might not be expanded due to discovered blocked cells. Additionally, the standard deviation for adaptive A* is 44,640, whereas standard deviation for forward A* is 44,741. There standard deviations for both are quite close, however Adaptive A* seems to be a more reliable search method due to its smaller standard deviation.

The closed list and goal cost from the previous search is saved and used in the current search to update heuristic values. When a node is first generated in the current A* planning, we check if it is in the last closed list, if it is then the heuristic is updated as:

$$h_{new}(n) = g(goal) - g(n)$$

where g(n) is the g(n) from the previous search. If the cell is not in the last closed list and was never generate in previous plannings, or if it the first planning, we update the h value to be Manhattan distances.