Analysis of A* Search

To talk about A* search, I will use the following notation:

- $g(s)$ denotes the cost from the initial state to state $s$.
- $h(s)$ denotes the heuristic estimate of the cost from state $s$ to the goal state.
- $h^*(s)$ denotes the cheapest path cost from state $s$ to a goal state.
- $f^*$ denotes the cheapest path cost from the initial state to a goal state.

**Theorem 1** If $0 \leq h(s) \leq h^*(s)$ for all states $s$, if the search graph is locally finite, and if each edge costs at least one unit, then A* search will find the optimal solution path.

**Proof:** For all states $s$ on the optimal path, $g(s) + h^*(s) = f^*$. This and $h(s) \leq h^*(s)$ imply that $g(s) + h(s) \leq f^*$ for all states $s$ on the optimal path. Thus, before A* halts, the priority queue will always contain some state from the optimal path with $g(s) + h(s) \leq f^*$.

Any state that is on the optimal path and in the priority queue will always be selected before any suboptimal goal state $s_{bad}$ because $g(s_{bad}) + h(s_{bad}) > f^*$. Therefore, A* search will visit the optimal path, including the optimal goal state, before any suboptimal goal state.

The locally finite and edge cost conditions guarantee a finite search.

**End Proof.**

To analyze the number of states that A* search visits, I make the following assumptions:

The search graph is a uniform search tree with branching factor $b$.

$0 \leq h(s) \leq h^*(s)$ for all states $s$, i.e., $h$ is admissible and never overestimates the true cost. By Theorem 1, this implies that A* search will find the optimal solution.

There is one goal state, which is distance $d$ from the initial state. [Having many goal states or many paths to a single goal state can hurt A* search because it might search all of them.]

All moves are reversible. This means the goal state is reachable from any state.

Each edge costs at least one unit. [To get similar results for IDA* search, I would need to assume that edge costs are positive integers.]

For all states $s$, $h^*(s) - h(s) \leq \epsilon$. I.e., there is some upper bound on the error.

**Lemma 2** Under the above assumptions, A* search will not visit any state that is more than $\epsilon/2$ distance from the optimal path.
Proof: Let $s_{bad}$ be a state that is more than $\epsilon/2$ away from the optimal path. Because each edge costs at least one unit, then $s_{bad}$ is at least $\epsilon/2$ cost away from the optimal path. Let $s_{good}$ be the state on the optimal path closest to $s_{bad}$. Then, $g(s_{bad}) > g(s_{good}) + \epsilon/2$ and $h^*(s_{bad}) > h^*(s_{good}) + \epsilon/2$. Then:

\[
g(s_{bad}) + h(s_{bad}) \geq g(s_{bad}) + h^*(s_{bad}) - \epsilon
\]
\[
> g(s_{good}) + \epsilon/2 + h^*(s_{good}) + \epsilon/2 - \epsilon
\]
\[
= g(s_{good}) + h^*(s_{good})
\]
\[
= f^*
\]

In the proof of Theorem 1, it was shown that if $0 \leq h(s) \leq h^*(s)$ for every state $s$, then every state $s_{good}$ on the optimal path has $g(s_{good}) + h(s_{good}) \leq f^*$, and that the priority queue always contains a state from the optimal path. So, because $g(s_{bad}) + h(s_{bad}) > f^*$, the goal state on the optimal path will be visited before $s_{bad}$.

End Proof.

The converse does not necessarily hold, i.e., it is not necessarily true that every node within $\epsilon/2$ distance of the optimal path will be searched, but this leads to a useful estimate of how many nodes $A^*$ might search. I.e., $A^*$ search will potentially examine every node that (1) is within $\epsilon/2$ distance of the optimal path, and (2) is on a level less than or equal to $d$.

Lemma 3 Suppose $\epsilon/2 \leq d$. Then, under the above assumptions, there are at most $db^{\epsilon/2} + 1$ states both within distance $\epsilon/2$ of the goal path and within distance $d$ of the initial state.

Proof: I prove the upper bound by mathematical induction.

Basis, $k = 0$, $l$ is any nonnegative integer. The only states that are both within distance 0 of the goal path and within distance $l$ of the initial state are the first $l + 1$ states on the goal path. Note that $lb^k + 1 = l + 1$ when $k = 0$.

Induction. Suppose that at most $lb^k + 1$ states are both within distance $k$ of the goal path ($k \geq 0$) and within distance $l$ of the initial state ($k \leq l$). Counting the children of these states counts all states within distance $k + 1$ of the goal path and distance $l + 1$ from the initial state, except that the initial state needs to be added back in. This results in:

\[
b(lb^k + 1) + 1 = lb^{k+1} + b + 1 = (l + 1)b^{k+1} - b^{k+1} + b + 1 \leq (l + 1)b^{k+1} + 1
\]

Thus, by mathematical induction, there are at most $lb^k + 1$ states both within distance $k$ of the goal path and within distance $l$ of the initial state. Using $k = \epsilon/2$ and $l = d$ results in $db^{\epsilon/2} + 1$ states, proving the lemma.

End Proof.

Lemma 2 and Lemma 3 prove the following theorem.

Theorem 4 Under the above assumptions, $A^*$ search will visit no more than $db^{\epsilon/2} + 1$ states.

The implication is that $A^*$'s running time is potentially (not necessarily) exponential in $\epsilon$, the amount that $h$ is in error. So, if $h$ is typically within 10% of the true value, you can expect the search time to be $O(db^{0.05d})$. This is a big improvement over blind search because this allows searches that are up to 20 times deeper (depending on how much time it takes to evaluate $h$). However, the order is still exponential and there will be some point where the “computational cliff” will take effect.