Notes on Optimal Search [1]

Optimal Search Methods

To obtain the pseudocode for the optimal search algorithms based on *branch-and-bound* we just need to make minor modifications to that for basic search.

Also, just as in informed/heuristic basic search, we can use a heuristic function $h(n)$ over the nodes n of the graph as an estimate of the cost to the goal from n.

There are different versions of branch-and-bound for optimal search depending on whether or not one uses a heuristic and whether one uses an extended list. The A^* *algorithm* is simply *branch-and-bound with a heuristic and an extended list.* To implement branch-and-bound using the pseudocode for basic search, we

1. add the extended paths anywhere in the queue, and
2. pick the partial path N with
   (a) the minimum cost of N, if we are *not using a heuristic*, or
   (b) the minimum sum of the cost of N and $h(\text{tail}(N))$, if we are *using a heuristic*.

To *guarantee* that branch-and-bound using just a heuristic (without an extended list) outputs an optimal path, it is *sufficient* that the heuristic values of the nodes in at least one optimal path do not overestimate their cost of the optimal path from the node to the goal. (Show that as an exercise.) The basic intuition behind this condition on the heuristic values is that they do not allow us to miss a shorter path to the goal because we have overestimated its costs due to an overestimate in one of the nodes in the optimal path. Since we do not know *a priori* what the optimal path is (that is precisely what we want to compute), we need to check that every node in a path to the goal satisfies that condition. A heuristic is *admissible* if for *every* node n in the graph, $h(n)$ never overestimates the cost of the optimal path from n to the goal.

However, using an admissible heuristic is not enough to guarantee that A^* will work; we actually need a stronger condition. To *guarantee* that A^* outputs an optimal path, it is *sufficient* that there exists at least one optimal path to the goal with nodes whose heuristic values satisfy the following condition: (*) if n and m are two consecutive nodes in the path and $c(n, m)$ is the cost of the (directed) edge from n to m, then $h(n) \leq c(n, m) + h(m)$. (See optional part of these notes for a proof—next section.) Once again, since we do not know the optimal path, we must check that every path to the goal satisfies that condition. A heuristic is *consistent* if condition (*) holds for every directed edge $n \rightarrow m$ in the graph.[2]

A consistent heuristic is admissible, but the opposite is not always true. (Show that as an exercise). Consistency is also easier to check than admissibility, since we only need to perform *local* checks for consistency (every directed edge), while admissibility generally requires a more *global* check (every path from each node to the goal).

[1] These notes are primarily based on the recitation notes of Kimberle Koile and the slides of Tomás Lozano-Perez from previous terms, as well as the books *Artificial Intelligence (Third Edition)* by Patrick Henry Winston and *Artificial Intelligence: A Modern Approach (Second Edition)* by Stuart Russell and Peter Norvig. Original date: September 24, 2004; Last updated: November 6, 2006.

[2] Note that an *undirected* edge in a graph is actually formed by two directed edges. Therefore, to check consistency, we need to check condition (*) in *both* directions.

On the Sufficiency of Consistency for A^* (Optional)

First, let me introduce some notation. Let s be the starting node and g be the goal. If X is a path such that $s \to \cdots \to y \to \cdots \to z \to \cdots \to g$ (*i.e.*, the path goes through y before z), let $c_X(y, z)$ be the cost of going from y to z along the path X. Naturally, if X is optimal, then $c_X(y, z) = c(y, z)$, where $c(y, z)$ is the minimum cost of any path from y to z. If X goes through y before going through z, we denote it by $z >_X y$. Similarly, for $y \leq_X z$. Finally, I say that a *path is consistent* if it satisfies condition (*) above. We denote the queue used by A^* by Q.

The result follows from the following two claims.

Claim 1 *For every node m in the search graph, if there is a consistent optimal path P to m, then for every node n in P, A^* would pick an optimal path to n from Q before a non-optimal path N to m.*

The claim follows because for every n in P,

$$\begin{aligned} c(s, n) + h(n) &\leq c(s, n) + c(n, m) + h(m) && \text{[since } P \text{ is consistent]} \\ &= c(s, m) + h(m) && \text{[by the definition of } c\text{]} \\ &< c_N(s, m) + h(m). && \text{[since } N \text{ is non-optimal]} \end{aligned}$$

The result now follows by Claim 1 if we can guarantee that an optimal path to a node in a consistent optimal complete path is always in Q. The following claim helps to provide such a guarantee.

Claim 2 *If there is a consistent optimal complete path O in the search graph, then the following invariant holds throughout the execution of A^*: if A^* picks an optimal path to some node $m \neq g$ in O from Q at round T, and for all rounds prior to T, there is a node $o \leq_O m$ such that Q has an optimal path to o, then Q has an optimal path to a node $n >_O m$ at round $T + 1$.*

The proof of this claim is as follows. If m has not been extended at round T, then the invariant holds. Otherwise, for the sake of obtaining a contradiction, assume Q does not have an optimal path to any $n >_O m$ at round T. Since, for all rounds prior to T, there is a node $o \leq_O m$ such that Q has an optimal path to o, then, by Claim 1, m must have been extended by some optimal path to an optimal path R_1 that ends at the node m_1 in O immediately following m. But since, by assumption, R_1 is not in Q at round T, we have that by Claim 1 again, m_1 must have been extended by an optimal path to an optimal path R_2 that ends at the node m_2 in O immediately following m_1. Continuing the same argument, we can conclude that Q had an optimal path to g at some round prior to T, but that optimal complete path is not in Q at round T. But this is a contradiction, since in that case A^* would have stopped prior to round T (that is, it could not pick R). Hence, the claim holds.

Since the condition in the invariant holds initially (*i.e.*, $m = s$, $T = 1$), applying Claim 2 recursively gives the result.