Properties of Heuristics that Guarantee A* Finds Optimal Paths

Robert Holte

this talk:
http://www.cs.ualberta.ca/~holte/CMPUT651/admissibility.ppt

Best-first Search

• Open list of nodes reached but not yet expanded
• Closed list of nodes that have been expanded
• Choose lowest cost node on Open list
• Add it to Closed, add its successors to Open
• Stop when Goal is first removed from Open

Dijkstra: cost, f(N) = g(N) = distance from start
A*: cost, f(N) = g(N) + h(N)
A* must re-open closed nodes

\[
\begin{array}{c}
\text{OPEN: (S,70)} \\
\text{CLOSED: (S,70)}
\end{array}
\]
A* must re-open closed nodes

OPEN: (A,120), (C,110)
CLOSED: (S,70), (B,40)

A* must re-open closed nodes

OPEN: (A,120), (D,110)
CLOSED: (S,70), (B,40), (C,110)
A* must re-open closed nodes

OPEN: (A,120), (G,140), (subtree with f=110)
CLOSED: (S,70), (B,40), (C,110), (D,110)

A* must re-open closed nodes

OPEN: (A,120), (G,140)
CLOSED: (S,70), (B,40), (C,110), (D,110), ...
A* must re-open closed nodes

**OPEN:** (G,140), (C,90)
**CLOSED:** (S,70), (B,40), (C,110), (D,110), …(A,120)

**Today’s Question**

When a node is first removed from Open, under what conditions are we guaranteed that this path to the node is optimal?

**Dijkstra:** all edge-weights are non-negative

**A**: the heuristic must have certain properties
Optimal Path to goal is the first off the Open list

S-N-G optimal, <N, g*(N)+h(N)> is on Open
<G,P> on Open is suboptimal
\[ g*(N)+h*(N) < P \]
\[ \Leftrightarrow h*(N) < P - g*(N) \]

Admissible Heuristic

Require <N, g*(N)+h(N)> lower cost than <G,P>
\[ g*(N)+h(N) < P \]
\[ \Leftrightarrow h(N) < P - g*(N) \]
\[ \Leftrightarrow h(N) \leq h*(N) \quad \text{(because h*(N) < P - g*(N))} \]

A heuristic is admissible if h(N) ≤ h*(N) for all N.
Admissible \( \Rightarrow \) first path to goal off Open is optimal
Optimal Path to X is the first off the Open list, for all X

S-N-X optimal, <N, g*(N)+h(N)> is on Open
<X,P+h(X)> on Open, P is suboptimal
\[ g^*(N) + c(N,X) < P \]
\[ \Leftrightarrow c(N,X) < P - g^*(N) \]

Consistent Heuristic

Require <N, g*(N)+h(N)> lower cost than <X,P+h(X)>
\[ g^*(N)+h(N) < P+h(X) \]
\[ \Leftrightarrow h(N) - h(X) < P - g^*(N) \]
\[ \Leftrightarrow h(N) - h(X) \leq c(N,X) \] (because \( c(N,X) < P - g^*(N) \))

A heuristic is consistent if \( h(N) \leq c(N,X) + h(X) \) for all X and all N.
Consistent \( \Rightarrow \) first path to X off Open is optimal for all X
Transforming heuristics into edge weights

Aim: replace the given edge weights and heuristics values with a set of edge weights (and NO heuristic) so that Dijkstra-costs on the new graph are identical to A*-costs on the given graph+heuristic

Transformation - goal

Dijkstra cost: h(S)  a+h(A)  a+b+h(B)
Transformation (1)

A* cost: \( h(S) \) \( a+\text{h}(A) \) \( a+b+\text{h}(B) \)

Dijkstra cost: \( h(S) \) \( a+\text{h}(A) \) \( a+b+\text{h}(B) \)

Transformation (2)

A* cost: \( h(S) \) \( a+\text{h}(A) \) \( a+b+\text{h}(B) \)

Dijkstra cost: \( h(S) \) \( a+\text{h}(A)-\text{h}(S) \) \( a+b+\text{h}(B) \)
Transformation (3)

A* cost: \( h(S) \) \( a+h(A) \) \( a+b+h(B) \)

Dijkstra cost: \( h(S) \) \( a+h(A) \) \( a+b+h(B) \)

Transformation - general

is transformed into

The order in which nodes come off the Open list using Dijkstra on the transformed graph is identical to the order using A* on the original graph+heuristic.
Local Consistency

If edge weights are non-negative, the first path to any node Z that Dijkstra takes off Open is an optimal path to Z.

Non-negative edge weights requires:
For all \( N \), and all successors, \( X \), of \( N \)
\[
0 \leq c(N, X) - h(N) + h(X) \\
\iff h(N) \leq c(N, X) + h(X)
\]

A heuristic is **locally consistent** if \( h(N) \leq c(N, X) + h(X) \) for all \( N \) and all successors \( X \) of \( N \).

Locally consistent \( \iff \) consistent

---

Monotonicity

With Dijkstra and non-negative edge weights, cost **cannot decrease** along a path since it is just the sum of the edge weights along the path.

Because A* with a consistent heuristic is equivalent to Dijkstra with non-negative edge weights, it follows that A* costs along a path can never decrease if the heuristic is consistent.

\[
A^* \text{ cost: } f(S) \leq f(A) \leq f(B)
\]
Admissibility $\not\Rightarrow$ Monotonicity

Along path S-A-C, f-values are not monotonic non-decreasing.

Enforced monotonicity

Can enforce monotonicity along a path by using parent’s f-value if it is greater than the child’s f-value.

(valid if h is admissible because the f values on a path never overestimate the path’s true length)

But this does not solve the problem of having to re-open closed nodes in our example.
Summary of definitions

- An admissible heuristic never overestimates distance to goal.
- A consistent heuristic obeys a kind of triangle inequality.
- With a locally consistent heuristic, h does not decrease faster than g increases.
- Monotonicity: costs along a path never decrease.

Summary of Positive Results

- Consistent $\Leftrightarrow$ locally consistent
- Consistent $\Rightarrow$ monotonicity
- Consistent $\Rightarrow$ admissible
- Consistent $\Rightarrow$ first path to X off Open is optimal, for all X
- Admissible $\Rightarrow$ first path to Goal off Open is optimal (correctness of the A* stopping condition)
Summary of Negative Results

• Admissible $\not\Rightarrow$ monotonicity

• Admissible $\not\Rightarrow$ consistent

• Admissible $\not\Rightarrow$ first path to $X$ off Open is optimal, for all $X$