Monotone Heuristics

• A monotone/consistent heuristic satisfies the triangle inequality: for all nodes n1, n2 and for all actions a
  \[ h(n1) \leq C(n1,a,n2) + h(n2) \]

Where \( C(n1, a, n2) \) means the cost of getting from the final state of n1 to the final state of n2 via action a.

• We showed in lecture that every monotone heuristic is admissible. The A* example from the tutorial verified that there exist admissible heuristics that are not monotone.
  • Hence monotone is a stronger condition than admissibility.
Monotone Heuristics

• Note that this definition applies to any path \( n \) and all of its successors.
• That is, if \( n \) is a path \(<s_0,\ldots,s_k>\) and the successor states of \( s_k \) are \( s_a, s_b, \) and \( s_c \) generated by the actions \( a, b \) and \( c \) respectively, then the paths
  \( n_a = <s_0,\ldots,s_k,s_a> \)
  \( n_b = <s_0,\ldots,s_k,s_b> \)
  \( n_c = <s_0,\ldots,s_k,s_c> \)
have the property that
  \( h(n) \leq \text{cost}(a) + h(n_a) \)
  \( h(n) \leq \text{cost}(b) + h(n_b) \)
  \( h(n) \leq \text{cost}(c) + h(n_c) \)
  when \( h \) is a monotone heuristic
Monotone Heuristics

- Consider a path $n = <s_0, s_1, ..., s_k>$ where action $a_i$ transforms $s_i$ to $s_{i+1}$. That is, $n$ is generated by the applying the action sequence $<a_0, ..., a_{k-1}>$ to the state $s_0$.

- $g(n)$ is the cost of the path $n$, so it is equal to the $\sum_{i=0}^{k-1} \text{cost}(a_i)$

- Consider all of the prefixes of $n$:
  $n_0 = <s_0>$
  $n_1 = <s_0, s_1>$
  ...
  $n_i = <s_0, s_1, ..., s_i>$
  $n_i+1 = <s_0, s_1, ..., s_{i+1}>$
  ...
  $n_k = <s_0, s_1, ..., s_k> = n$
Monotone Heuristics

• $n_0 = <s_0>$
  $n_1 = <s_0, s_1>$
  ...
  $n_i = <s_0, s_1, ..., s_i>$
  $n_{i+1} = <s_0, s_1, ..., s_{i+1}>$
  ...
  $n_k = <s_0, s_1, ..., s_k> = n$

• We can observe that
  $h(n_0) \leq \text{cost}(a_0) + h(n_1)$
  $h(n_1) \leq \text{cost}(a_1) + h(n_2)$
  ...
  $h(n_{k-1}) \leq \text{cost}(a_{k-1}) + h(n_k)$

• Therefore
  $f(n_0) = g(n_0) + h(n_0) \leq$
  $\leq \text{cost}(a_0) + h(n_1) =$
  $g(n_1) + h(n_1) = f(n_1)$
  $f(n_1) = g(n_1) + h(n_1)$
  $\leq g(n_1) + \text{cost}(a_1) + h(n_2) =$
  $g(n_2) + h(n_2) = f(n_2)$
  ...
  $f(n_{k-1}) \leq f(n_k)$
Monotone Heuristics

- In other words we have proved the following result: Proposition: with a monotone heuristic the f-values along a path are non-decreasing.

- Where by f-value along a path we mean the f-values of successive prefixes of the path.
Monotone Heuristics

• With this we can prove the following

**Proposition:** The sequence of f-values of the nodes expanded by A* using a monotone heuristic are non-decreasing.

**Proof:** We prove this by induction on k the number of nodes expanded by A* since it started.

**Base case:** $k = 0$. A* has not expanded any nodes, and so the sequence of f-values is empty and thus trivially non-decreasing.
Monotone Heuristics

- **Proposition:** The sequence of f-values of the nodes expanded by A* using a monotone heuristic are non-decreasing.

**Proof:**

**Induction.** Assuming proposition is true after A* has expanded k nodes, we prove that it is true after A* expands k+1 nodes.

Let $n_{k+1}$ be the node expanded by A* at step k+1, and $n_k$ be the node expanded by A* at step k. We must show that $f(n_k) \leq f(n_{k+1})$. We have two cases

a) $n_{k+1}$ was on OPEN at step k. A* expands the node on open with lowest f-value, and it choose to expand $n_k$ at step k. Hence $f(n_k) \leq f(n_{k+1})$

b) $n_{k+1}$ was on not OPEN at step k. So $n_{k+1}$ must have been added to OPEN when A* expanded expand $n_k$ at step k. Hence, $n_k$ is a prefix of $n_{k+1}$ and by the previous proposition $f(n_k) \leq f(n_{k+1})$