0.1 A* Evaluation Function

$$f(n) = g(\text{State}_n) + h(\text{State}_n)$$

where:

- $g(n)$ – initial node

0.2 Admissible Heuristic: Proof

An admissible heuristics $h()$ is guaranteed to give you the optimal solution. Why? Proof by contradiction:

- Say: the algorithm returned a suboptimal path ($C > C*$)
- So: there exists a node n on $C*$ not expanded on C:

If so:

$$\begin{aligned} &f(n) > C* \\ &f(n) = g(n) + h(n) && \text{(by definition)} \\ &f(n) = g*(n) + h(n) && \text{(because } n \text{ is on } C*) \\ &f(n) \leq g*(n) + h*(n) && \text{(if } h(n) \text{ admissible: } h(n) \leq h*(n)) \end{aligned}$$

0.3 What Made A* Work Well?

- Straight-line heuristics is consistent: its estimate is getting better and better as we get closer to the goal
- Every consistent heuristics is admissible heuristics, but not the other way around

But that would mean that:

$$f(n) \leq C*$$

0.4 A*: Search Contours

How does A* "direct" the search progress?

0.5 Dominating Heuristics

We can have more than one available heuristics. For example $h_1(n)$ and $h_2(n)$. $h_2(n)$ dominates $h_1(n)$ iff[1] $h_2(n) > h_1(n)$ for every n.
If you have multiple admissible heuristics where none dominates the other:

$$\text{Let } h(n) = \max\left(h_1(n), h_2(n), \ldots, h_m(n)\right)$$

[1] if and only if

0.6 Domination $\rightarrow$ Efficiency: Why?

With

$$f(n) < C* \text{ and } f(n) = g(\text{State}_n) + h(\text{State}_n),$$

we get

0.7 Domination $\rightarrow$ Efficiency: But?

If $h_2(n)$ dominates $h_1(n)$, should you always use $h_2(n)$? Generally yes, but $h_2(n)$ vs $h_1(n)$ heuristic *computation time* may be a deciding factor here.

0.8 Heuristic and Search Performance

- Consider an 8-puzzle game and two admissible heuristics:
  - $h_1(n)$ – number of misplaced tiles (not counting blank)
  - $h_2(n)$ – Manhattan distance

0.9 $h()$ Quality: Effective Branching

0.10 Can We Make A* Even Faster? (Sometimes at a cost!)

0.11 Weighted A* Evaluation Function

$$f(n) = g(\text{State}_n) + W * h(\text{State}_n)$$

where:

- $g(n)$ – initial node to node n path cost
- $h(n)$ – estimated cost of the best path that continues from node n to a goal node
- $W > 1$

Here, weight W makes $h(\text{State}_n)$ (perhaps only "sometimes") inadmissible. It becomes potentially more accurate = less expansions!