# CSCE625: Artificial Intelligence
# Sample Midterm Exam Questions (Solutions)

## Question 1: Heuristics (15 points)

Consider heuristics where $h(n) = 0$ for all $n$ for which $\text{GOAL}(n) = \text{True}$, i.e., functions that have the value zero for states that are goals. Then the following statements are either true or false

**Statement 1:** Every admissible heuristic is consistent.

**Statement 2:** Every consistent heuristic is admissible.

For each statement: (1) identify whether it is true or false. (2) If it is true prove that it is so; otherwise, if it is false, provide a counterexample.

> *Reminder:* Using the notation from the textbook, with $n'$ a successor of state $n$ and $a$ an action, a consistent heuristic is required to have $h(n) \leq c(n, a, n') + h(n')$.

Statement 1 is *false*. Here is a counter-example. Assume a linear graph between these cities, with the numbers appearing above as true costs $A \xrightarrow{10} B \xrightarrow{10} C \xrightarrow{10} D \xrightarrow{10} Goal$.
Now consider the heuristic defined thusly:

$$h(D) = 9 < 10$$
$$h(C) = 19 < 20$$
$$h(B) = 2 < 30$$

Statement 2 is *true*. Here is a proof by contradiction (though, induction would work as well):
Assume the contrary and consider node $n$, the one of least distance from goal that violates admissibility. Thus,

$$\text{true\_cost\_to\_goal}(n) < h(n).$$

Now, $n$ can't be a goal itself, since $h(n) = 0$ is admissible. So $n$ must be connected, on its least cost path, to some other node *en route* the goal. Call that node $n'$. Then, since step costs are positive, we see

$$\text{true\_cost\_to\_goal}(n') < \text{true\_cost\_to\_goal}(n).$$

Also, $h(n') \leq \text{true\_cost\_to\_goal}(n)$, and accordingly,

$$\begin{aligned} h(n) &\leq c(n, a, n') + h(n') \\ &\leq c(n, a, n') + \text{true\_cost\_to\_goal}(n') = \text{true\_cost\_to\_goal}(n), \end{aligned}$$

but that means $h(n) < h(n)$, which is contradiction.

## Question 2: Search costs (6 points)

You are approached by a client who arranges cycling tour holidays of Romania; they are interested in a specialized routing problem on their map, which they want to treat via search on a graph. Usually when one looks for a route to drive from one place (say Arad) to another (say Bucharest), the quantity being minimized is the total distance. If the route goes from $A$ to $S$, on to $F$, then arrives at $B$, the total distance is the sum of distances on the roads along the path (i.e., $\text{dist}(A, S) + \text{dist}(S, F) + \text{dist}(F, B)$). But your client is thinking about how grueling a route will be, so they have boiled their problem down to minimizing of either:

1. the mean distance between stops (e.g., $\frac{1}{3}\left[\text{dist}(A, S) + \text{dist}(S, F) + \text{dist}(F, B)\right]$) or,
2. the maximal leg distance (e.g., $\max[\text{dist}(A, S), \text{dist}(S, F), \text{dist}(F, B)]$).

Assuming that you want to use uniform cost search, what do you advise them? Why? Argue about the metrics w.r.t. the algorithm, not which is a better model of bicycling route difficulty.

Tell them to use the maximal leg distance measure.

- The mean distance measure is a bad idea because it is not monotonic. If the first leg has distance $10$ then the second has distance $5$ and the third $2$, moving along the path gives scores of $10$, $\frac{1}{2}(10 + 5) = 7\frac{1}{2}$, and $\frac{1}{3}(10 + 5 + 2) = 5\frac{2}{3}$. Attempting to use UCS will cause problems, as this is akin to using the normal item with negative weights. Going round and round cycles in the graph can decrease the mean distance.

- Using the maximal leg distance is very easy to do with minimal modification: replace the summation with a $\max(\cdot)$ in your program. This can be done incrementally the way the algorithm already treats it since, for example,

$$\max[\{v_0, v_1, v_2, v_3\}] = \max\left(\max\left(\max\left(v_0, v_1\right), v_2\right), v_3\right).$$