**AI – Heuristic and informed search – 04**

***Heuristic* h(n)**

A heuristic is a function h(n) is a guess to produce a feasible solution in a reasonable time span

**h(n) = estimated cost of the cheapest path from the state at node n to a goal state**

This means that we are trading off on solution optimality (e.g. the shortest path), but we rather gain on time, and **a** solution is guaranteed to be found

***Best-first search (example of informed search)***

Best-first search refers to a **class** of search algorithms, in which the most promising (according to predefined rules) is chosen to be explored first, usually implying the usage of **distance as heuristic** via setting a **priority queue**

To search the graph space, the BFS\* method uses two lists for tracking the traversal:

* + An ‘Open’ list that keeps track of the current ‘immediate’ nodes available for traversal
  + A ‘Closed’ list that keeps track of the nodes already traversed.

A natural question would be “**how does this differ from Dijkstra’s algorithm?”**

BFS\* fails on weighted graphs, as it will choose **distance** over **cost of edge traversal**

***Greedy best-first search***

Greedy BFS\* will always expand to the most promising node (i.e. nearest), as said before, the example in the slides uses **straight line distance** (which is, trivially, **less** than the actual distance)as heuristic for the example, and it certainly finds a solution, just not the best

This is because the usage of backtracking is not considered, as Greedy BFS\* focuses on exploring the most promising path **without considering alternative paths**

It can lead to dead ends and/or be suboptimal, and will need backtracking if we want more out of it

(Need to ask this)

***A\* algorithm***

**Optimal for positive costs**

A\* algo comes from the fact that heuristic is often misleading, by having the **evaluation function** (aka fitness number):