A^*

↳ Heuristic Function

↳ Estimates cost Reaching Goal from a Node,

↳ Considers both Distance already traveled from Start And Remaining Distance to the goal.

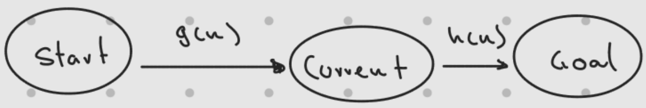
Concepts

↳ Node: Point in your Graph

↳ Edges: Connection between Nodes

- Path cost: Cost of moving from Node to Node
- Heuristic: Estimated Cost from Any Node to Goal
- Search space: Collection all poss. paths.

$f(n) = g(n) + h(n)$



Cost function:

- Path cost G(n):

↳ Exact known Distance from Initial Node

↳ How to Calculate? $\underbrace{\sum_{i=0}^{n-1} w(n_i, n_{i+1})}_{\text{weight of Edge connecting Nodes}}$

$$g(n_n) = \sum_{i=\emptyset}^{n-1} w(n_i, n_{i+1})$$

Accumulates as we move through Graph.

- Heuristic function h(n):

↳ Estimated Cost from current node to goal node

↳ For any Node, the heuristic estimate must satisfy: $h(n) \leq h^*(n)$

↳ Actual cost to Goal.

→ to never overestimate

- For MAPS

↳ Manhattan Distance:

- $h(n) = |x_1 - x_2| + |y_1 - y_2|$

↳ No se puede mover en Diagonal

↳ Euclidean Distance

$h(n) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$

↳ se usa para Robots que pueden ir en cualquier Dirección!

Total Estimated Cost:

$f(n) = g(n) + h(n)$ → h(n) es Importante ya que podría tomar Caminos muy largos que parecen prometedores

Things to code:

↳ Open list:

- Nodes to be evaluated
- Sorted by $f(n)$ value
- New nodes Added as Discovered

↳ Closed List:

- Evaluated Nodes
- Avoid re-evaluating nodes
- Reconstruct final path