Task 15 – Spike: Goal-Oriented Action Planning (GOAP)

Context:

Goal Oriented Action Planning (GOAP) is an extension to simple goal insistence (SGI) action planning that considered the outcome of actions and creates plans instead of simple reactions to goal insistence levels.

Knowledge/Skill Gap:

Developers need to be able to create and use a GOAP system for agent control.

Goals/Deliverables:

Create a GOAP simulation that demonstrates the effectiveness of the technique in considering long-term outcomes of actions (related to side-effects and/or time delays) and can plan and act intelligently.

Start-End Period: Week 9 – Week 10

Planning Notes:

Time to create your own...

Extensions:

Think of your own...

Making Cost-Based Searches Work Correctly.

There are three main things to change (but you might want to do more):

- In the box_world.py module there is a variable named min_edge_cost used by the A* algorithm. It is currently set to the wrong value! Change this value and demonstrate (find a case) where the behaviour between the old value (which can give non-optimal results) and your new value (which should give optimal results) are different. The console will show the path cost to assist in your comparisons.

- Currently the navigation graph only uses N-S-E-W edges between boxes. A* is also currently using Manhattan distance to estimate cost. Find the code in the reset_navgraph method of the BoxWorld class (in the box_world.py module) and uncomment the code needed to add diagonal edges.

- Is the use of Manhattan distance calculation correct now? Change this to another method by editing the reset_navgraph method of the BoxWorld class. (The correct code is currently commented out.)

By the end of this lab you should have a solid understanding of the differences between each of the search methods used, and be able to describe the behaviour of each. In particular you must understand the influence of the edge_cost_value on A* star search, and the different types of cost estimation heuristics (and how they must match the topology and the real cost in some way.

Optional Steps

- Create your own maps.
- Add additional box types, costs and colours.
- Create an agent the follows each node of the path in order (.

All Keys

There are many other features that can be changed, and some keys have been assigned:

1: “clear” (white)
2: “mud” (grey-brownish)
3: “water” (blue)
4: “wall” (black)
5: “start”
6: “target”
SPACE: force a replan.
N and M: cycle (back or forward) through the search methods and replan.
UP and DOWN: Increase or decrease the search step limit by one.
0: Remove the search step limit. (A full search will be performed.)

E: toggle edges on/off for the current navigation graph (thin blue lines)
L: toggle box (node) index values on/off (useful for understanding search and path details).
C: toggle box “centre” markers on/off
T: toggle tree on/off for the current search if available
P: toggle path on/off for the current search if there is a successful route.