Juan Barrera

Professor Mark Voortman

Introduction to AI

Module 2 Assignment (Part A)

Sep 15 20205

Questions:

1. Why problem formulation must follow goal formulation:

We can understand problem formulation as knowing what we want to achieve, following this logic we can say that: Understanding goal formulation specifies how we are going to know we have suceeded like for example making our mouse reach the exit for a maze. Problem formulation specifies the search for the solution, here we day what steps are necessary to achieve our goal. Knowing the goal first help us know what details are really relevant, it goes first so that way, the representation of the problem is not too detailed yet but neither too abstract, we start with an idea and work on that.

2. In navigating a maze, the only place we need to turn is at the intersection of two or more corridors. Reformulate this problem using this observation. How large is the state space now?

In the observation we can see that we only need to decide at intersections (junctions, turns, dead-ends), not every cell.

So we can say the following:

States: Robot is at a particular intersection, facing a certain direction. If there are I intersections in the maze:

State Space Size = $4 \times I$

State Space Size=$4\times$I

This is much smaller because

I$\ll$N (fewer intersections than individual squares).

3. From each point in the maze, we can move in any of the four directions until we reach a turning point, and this is the only action we need to do. Reformulate the problem using these actions. Do we need to keep track of the robot's orientation now?

Now we can define actions as: "Move north until stop" then "Move east until stop," etc.

With this reformulation:

Now: The robot is at a turning point (position only).

The actions required are: From each point, you have up to 4 possible moves (to the next turning point in each direction).

Orientation is no longer needed because each action will take it directly to the next turning point so there is no intermediate state that depends on which way we are facing.

Then it is simply:

State Space Size=I

4. In our initial description of the problem we already abstracted from the real world, restricting actions and removing details. List three such simplifications we made.

Ignored continuous space:we segmented the maze into grid squares and intersections.

Ignored dynamics:no acceleration, inertia, or slippage considered.

Perfect sensing and actuation:assumed that the robot always knew the exact position and orientation, and actions succeed perfectly.

In my own words:

Term Meaning

State: A description of the world at a given point (e.g., robot's position and orientation).

State Space: The set of all possible states you could be in.

Search Tree: A tree data structure built as we explore states: root is start state, branches are actions, nodes are resulting states.

Search Node: A record in the search tree containing a state, a pointer to its parent, the action that led here, and possibly path cost.

Goal: A condition or set of states that satisfy success criteria.

Action: A possible step the agent can take from a given state.

Transition Model: A description of how each action leads to a new state.

Branching Factor: The average number of successors (actions) per state in the search tree.

Difference between a world state, a state description, and a search node? Why is this distinction useful?

A world state is the complete, real configuration of the environment, the actual physical situation as it exists in reality. A state description is an abstraction of that world state, representing just those features relevant to the problem solving activity, such as a robot's location and orientation within the maze. A search node refers to the structure that is used during the search and contains a state description, along with additional information such as the parent node, action leading to the node, and cost for the particular path. The distinction is helpful because it separates reality from our representation thereof and from the bookkeeping mechanism required for search. Consequently, it allows us to model only those aspects not considered relevant to the problem, thus ensuring that the search algorithm will find solutions efficiently by searching through manageable abstractions rather than through the messy world itself. Also, paths and costs can be reconstructed without complicating the actual state space.