Lab session 3: Searching in more difficult tasks

Pierre-Alexandre Murena

April 12, 2024

This session is divided into three exercises:

- **Exercise 1** proposes a direct application of the notions seen in the lecture for search with non-deterministic actions.
- **Exercise 2** is an open problem involving search. The first part of the exercise (problem formulation) will be done in small groups; the second part (implementation) is optional. The code can be sent to us for a bonus of 3 points on your final grade[1].
- **Exercise 3** is a step-by-step construction of a belief-state search problem.

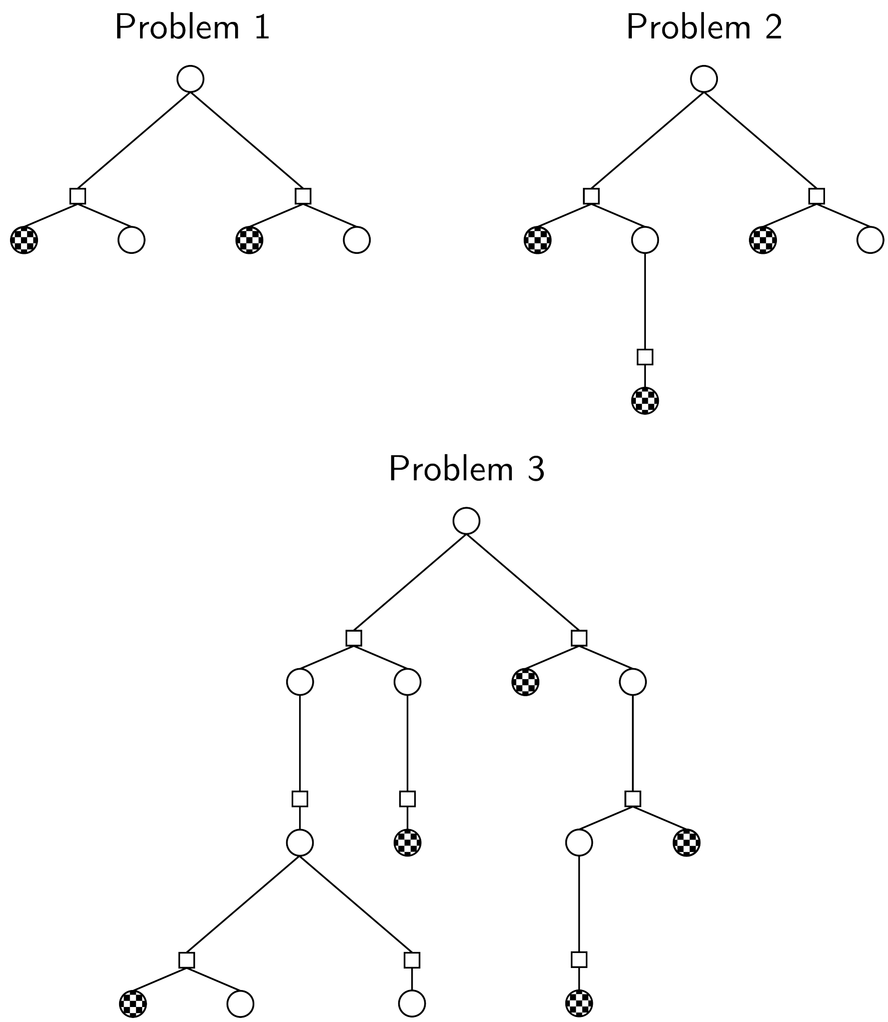
GitHub repository: `https://github.com/ppaamm/TUHH-AICourse`

[1] As a reminder, the final grade is on 100 points (30 pts for the graded lab session + 70 pts for the written exam). 50 points are required to pass. Bonus points are simply added to the total number of points.

1 Direct applications

The purpose of these exercises is to properly understand the notions related to search with non-deterministic actions.

Question 1. Do the following search problems (represented as AND-OR graphs) have a solution? If yes, describe it.



2 The robot guide and the starving tourist

In this exercise, we study a robot-assistant whose goal is to guide a tourist to reach a good restaurant. The city is known for having two good restaurants, but also three restaurants that are frauds for tourists. The location of the restaurants is shown on the map in Figure 1.

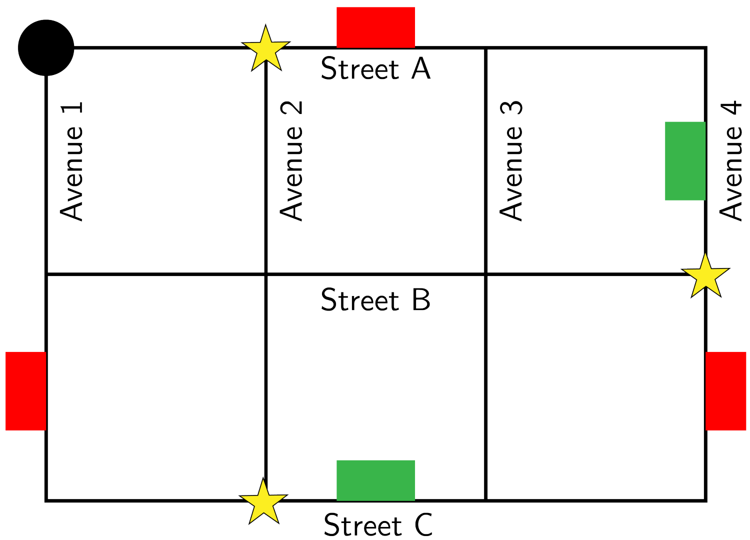


Figure 1: Map of the city. Black circle: initial position. Red rectangle: fraud restaurant. Green rectangle: good restaurant (goal). Star: stunning intersection.

The task of the robot-assistant is to suggest a direction to follow at each intersection. The tourist will always follow the instructions of the guide, except at stunning intersections. Stunning intersections are so beautiful that the tourist might ignore the recommendation and just explore any of the streets. The robot must be careful, because the tourist is particularly hungry: they will enter the first restaurant they pass by.

2.1 Problem formulation

Question 2. Express this problem in the form of a well-specified search problem. Is this search with deterministic or non-deterministic actions? Is it fully or partially observable?

Question 3. Write a corresponding search graph, or AND-OR graph in case the task is non-deterministic.

Question 4. What are the solutions of the problem?

2.2 Implementation of the search (Bonus, 3 pts)

In the file `city.py`, you can find an implementation of an AND-OR tree, as well as an implementation of the environment. (A good exercise would be to code your own version of the environment by yourself)

The `and_or_dfs` method proposes a recursive version of the AND-OR graph search. As seen during the lecture, it relies on two sub-methods: one for the AND nodes and one for the OR nodes. The search for the OR nodes is the same as in the standard DFS.

Question 5. Implement the search for the AND nodes. Execute the search on the guiding task. What is the plan to reach a good restaurant?

Question 6 (Difficult)**.** If you launch a search starting from the intersection of Street C and Avenue 1, you will notice that the search fails and the program does not find a solution. Identify the problem (It is not expected that you fix it[2]).

3 Searching with partial observations

In this exercise, we will consider the vacuum cleaner environment, in two variants: sensorless and partial observability.

3.1 The vacuum cleaner environment

The environment consists of two cells (one at the left, one at the right). Each cell can be either clean or dirty. The agent is located in one cell and has three possible actions:

- RIGHT: Moves to the right cell
- LEFT: Moves to the left cell
- CLEAN: Cleans the dust in the current cell (if any)

The goal of the agent is to make sure that all cells are clean.

Question 7. List all possible states in this environment.

3.2 Sensorless vacuum cleaner

We now consider that the agent sees *nothing* about the environment: it means that it doesn't know its current position nor the state of the cells.

Question 8. Without any further information, in which state can the agent be initially?

A set of (physical) states is called a **belief-state**. When the agent does not know the exact state, it makes hypotheses about the actual state. When an agent has a given belief state, it means that it does not have evidence to discriminate any of the states. In our example, at the beginning, the agent has no information, so cannot tell which of the states it is in.

Question 9. From there, the agent plays action LEFT. In which states can it be now? Then it plays CLEAN. Where can it be now?

[2]If you do, you will obviously get an extra-point. But fixing this problem is extremely difficult, you are warned! Try it at your own risks.

This corresponds to the next belief state after playing action Left. Such as for standard states, you can create a tree of belief states, where a branch from belief state i to belief state j means that you can reach belief state j from belief state i by playing some action.

Question 10 (Optional)**.** Draw the full search graph on belief states.

3.3 Vacuum cleaner with partial observability

We now consider that the agent is able to see only the content of its cell. In particular it does not see the content of the neighboring cell nor its position.

Question 11. Initially, the agent observes that it is in a clean cell. In which state can the agent be initially? This set of states is the initial belief state.

Question 12. From there, the agent plays action Left. In which states can it expect to be now?

Question 13. Now that it is in this new (physical)[3] state, the agent observes the presence of dust in this new cell. In which states can it be now?

[3]The agent is indeed in a physical state, it just cannot observe it. The belief state represents what the agent thinks, based on what it observed.