Fundamentals of Artificial Intelligence

Programming Exercise 3: Logic 16th December, 2022

Yuanfei Lin, Adrian Kulmburg (last updated 16th December 2022)
Oliver Hausdoerfer, Yikai Jin (in alphabetical order by last name)

Problem 3: Cleanup-Day

Problem Description

Keeping the local parks in urban regions - such as the Olympiapark in Munich - free from litter is an important task to provide the communities with enjoyable recreational areas. Although there are existing initiatives, like the *Global Cleanup Day* or more local campaigns such as *Cleanup Munich*, in our fictive scenario the municipality has decided it needs further help with this problem. An autonomous robot - '**EcoBot**' - capable of navigating through the park and cleaning litter on the way should be designed.

The goal of this exercise is to program 'EcoBot' such that it can autonomously navigate through the provided park layouts and clean litter on its way. This is achieved by filling a knowledge base with logical sentences. More details regarding the implementation and what is ask of you is provided in the **notebook** for this problem, which you installed from Artemis[1].

Have fun cleaning our virtual parks and keep nature beautiful!

Passing the Exercise

You will have passed the exercise if you can successfully solve all 5 scenarios that are given to you together with the notebook. Additionally, your code also has to solve 3 hidden scenarios, that are only available on Artemis.

If your implemented function computes the knowledge base such that every scenario is solved correctly (including the hidden ones), you successfully completed this programming exercise.

Your code has to compute a valid solution for each of the scenarios within 5 min on our machine. If your code takes longer to compute a solution, you will fail this submission. Don't worry about the computation time too much as usually the algorithm produces a solution within seconds for our specific exercise. Your submission will be evaluated after the deadline, but until then you can update your solution as many times as you like. The last submitted solution will be graded.

Programming Framework

For this programming exercise, a *Jupyter Notebook* will be used. To model the logic problem, you should be familiar with Python. The main function of the template is in the **logic_exercise.ipynb** file, which is also the only file you have to work on.

[1]https://artemis.ase.in.tum.de/courses/222/exercises

The following steps are required to correctly set up the environment for the programming exercise and submission:

1. **Installation of AIMA**: Work through AIMA installation instructions on Moodle[2]. (Using Docker is recommended for beginners.)
2. **ARTEMIS**: Log into ARTEMIS with your TUM credentials. Find the exercise *Logic* and follow the installation and submission instructions.

Inference Algorithm

The inference algorithm for this exercise uses the *Davis-Putnam-Logemann-Loveland (DPLL)* algorithm, which is not part of the lecture. It is a backtracking algorithm for inference in propositional logic, and functions in a similar way as the backwards-chaining algorithm discussed in the lecture. If you are curious to discover how the algorithm works, you can check out e.g., `https://en.wikipedia.org/wiki/DPLL_algorithm`. The algorithm is able to handle knowledge bases with arbitrary structure (i.e., it does not require Horn-clauses), and is therefore recommended for this exercise.

Submissions will close on **27th January at 23:59**. Your solution will be graded by ARTEMIS. There will be feedback on formatting errors and rightly solved maps. Nonetheless, it is very important to follow the instructions exactly!

ATTENTION

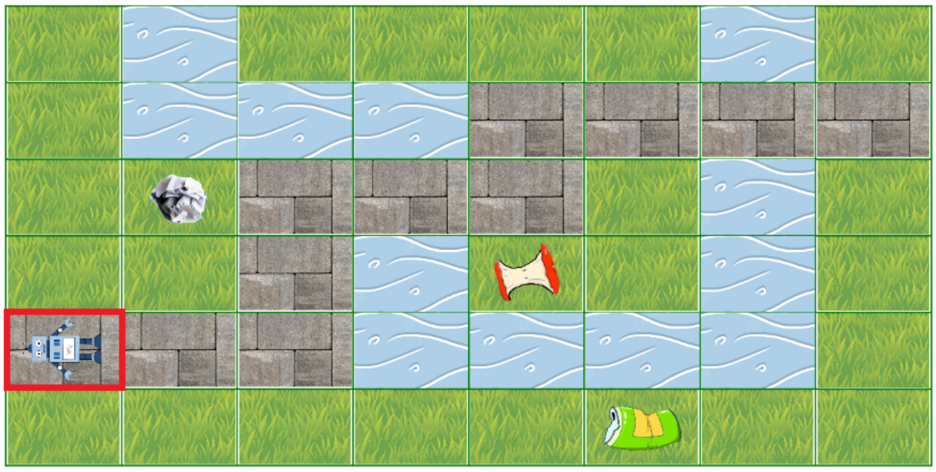
- Your code should **not** print anything.
- **Do NOT** rename the submitted file or the function name. If you do you will fail!
- **Do NOT** import any additional modules for your solutions. If you do you will fail!
- Like the rest of the programming exercises, this is an individual project and work **must** be your own. We will use a plagiarism detection tool and any copied code will cancel all bonus points from programming exercises for both the copier and the copied person!

[2]https://www.moodle.tum.de/mod/page/view.php?id=2323882

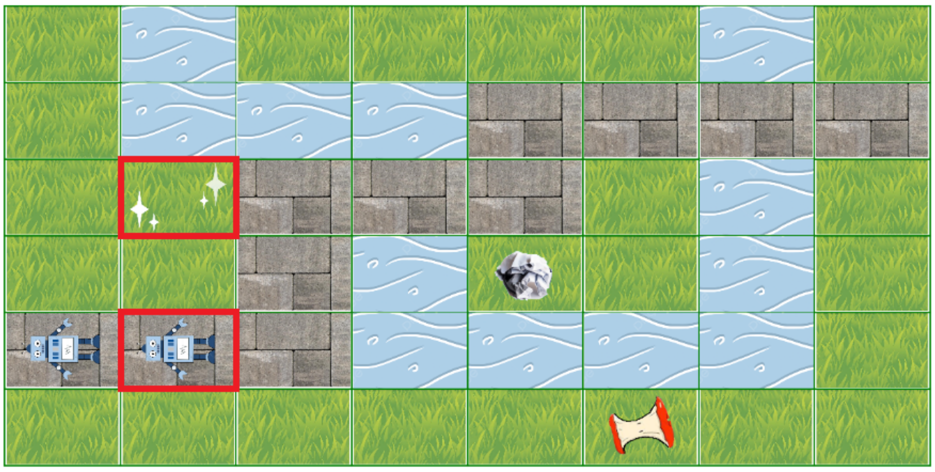
***Sample Solution for the scenario Nr. 3**

Here, for better understanding of the problem, the correct solution for scenario 3 provided with the notebook is given. The solution for this scenario inferred by your knowledge base should look exactly like the solution provided. Each tile mentioned in the explanation is additionally marked with a red rectangle in the pictures for better orientation. The red rectangle will not be part of the output of your final solution. Please make sure to check out the instructions in **logic_exercise.ipynb** first, before trying to understand this solution.

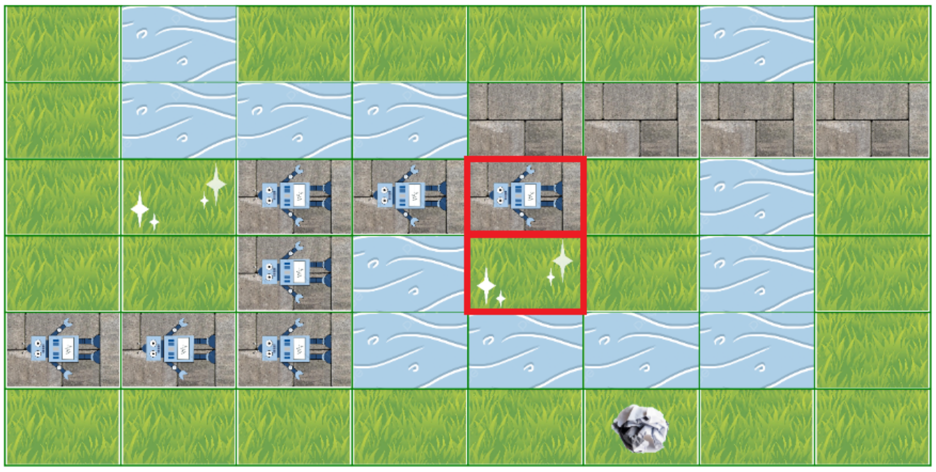
Iteration 0: 'EcoBot' is shown in its starting position (4,0) on the map. The tile (4,0) is marked as visited 'V' in the knowledge base.



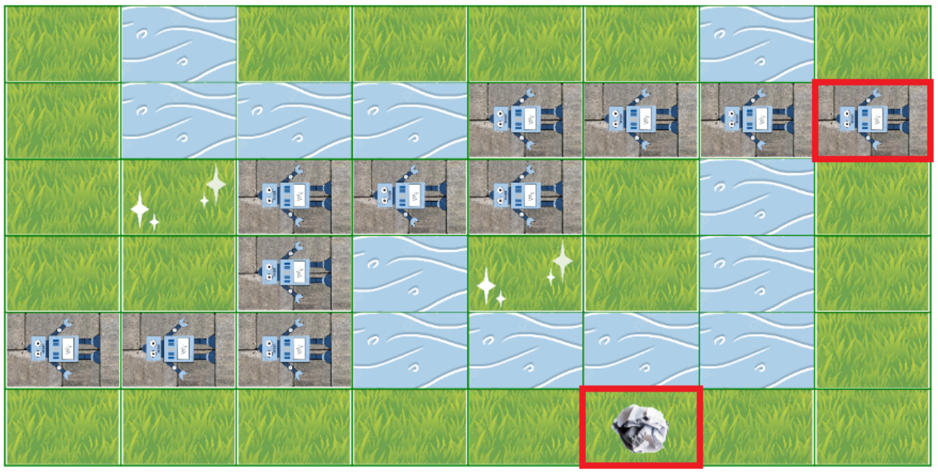
Iteration 1: 'EcoBot' advances to tile (4, 1), because there is reachable litter in column 1 at tile (2,1). The litter is then cleaned automatically. Each visited tile on the path is marked as visited 'V' in the knowledge base.



Iteration 2: 'EcoBot' advances to tile (2, 4), because there is reachable litter in column 4 at tile (3,4). Again, the litter is cleaned automatically and each tile on the path visited by 'EcoBot' is marked as 'V' in the knowledge base.



Iteration 3: 'EcoBot' advances until the end of the pathway (1, 7), because there is no more reachable litter on its way. *Note: the litter on tile (5, 5) is not reachable, as it is located behind a river.*



With this last iteration, the scenario is solved correctly.