Multi-Agent Systems

B. Nebel, F. Lindner, T. Engesser
Winter Semester 2018/19

University of Freiburg
Department of Computer Science

Exercise Sheet 9

Due: January 9, 2019, 16:00

Exercise 9.1 (DominationWorld I, 4+2)

In this and the following exercises, we want to implement a BDI agent for a domination-style setting: The task is to continuously activate and control as many checkpoints as possible on a grid world. Once a checkpoint is activated, it will deactivate automatically after a fixed period of time (30 seconds). For each active checkpoint, the team receives one point per second. Important tasks of the agents involve exploration of the map, division of the checkpoints among the agents, and continuous (re-)activation of the checkpoints. You can find the template for this project in the course's ownCloud folder (the place where you can also find the solution slides). If you don't know the URL, please send a mail to `engesser@cs.uni-freiburg.de`. Both the simulation and the GOAL agent specification are adapted from the original GOAL example project VacuumBots.

(a) Design (and implement) the knowledge base of your agents representing the observed environment. This representation should include knowledge and beliefs beyond current perception, i.e., about entities seen some cycles before. Note that the current coordinates of the agent, as well as its orientation are available as percepts. (Hint: you can see all available percepts when running the simulation in debug mode.)

(b) Explain how you plan to use your knowledge base to inform your agents' practical reasoning. Which roles do beliefs, desires, and intentions play?

Exercise 9.2 (DominationWorld II, 4+2)

Your task in this exercise is to further equip the DominationWorld agents both with the ability to find optimal paths and with a strategy to explore the map.

(a) Implement the agents' ability to identify optimal paths from arbitrary accessible starting cells to arbitrary accessible destination cells on the map. To achieve this, you can, e.g., define a new predicate `dist(Xstart, Ystart, Xdest, Ydest, D)` in Prolog which tells you that with the information from the knowledge base, the shortest known path between the coordinates `Xstart`, `Ystart` and `Xgoal`, `Ygoal` has distance `D`. *Hint*: A simple definition works by induction over `D`: For arbitrary coordinates `X`, `Y` we have `dist(X, Y, X, Y, 0)`. If `X2`, `Y2` is reachable from `X`, `Y` with distance `D` then all neighbors of `X2`, `Y2` are reachable from `X`, `Y` at least with distance `D+1`.

(b) Implement a reasonable exploration strategy. Your agents should always be aware of areas on the map that have still to be explored, if there are any. It might be useful to calculate optimal paths to fields that you plan to explore. To avoid agents blocking each other indefinitely you might want them to perform random actions from time to time. The goal of your agents is to completely explore the accessible part of the map. Afterwards, the agents should stop.

Please export your Eclipse project as an archive file and submit it to `engesser@cs.uni-freiburg.de`.