Assignment 4
v1.3

April 2020

1 Introduction

This exercise aims at knowledge representation, reasoning, and planning in first-order logic. You will apply situation calculus in order to plan in a dynamic task in terms of first-order logical deductions.

2 Introductory minitask

The goal of this minitask is to provide a lighter introduction to situational calculus.

2.1 Problem description

You have been provided a small world with situational calculus where there are two agents (`car` and `bike`) and only one action possible (`move`) in the file `introductory_assignment_template.txt`. Your goal is to verify the behavior of the world by writing several conjectures that the prover attempts to prove. The conjectures are specified in the template of the minitask and have to be named as `<username>_conjecture_XX`, where `<username>` is your username and `XX` is the conjecture number — there are two conjectures provided as examples.

3 Main task

3.1 Problem description

The task is simple — Alice wants to obtain a treasure that is in the *room D*. However, Alice is in the *room A* and there is a locked door (in Fig. 1 shown in grey) between her and the treasure. She needs Bob to help her. Bob is in the *room B* where is also the key from the locked door. He needs to pick up the key and open the door so that Alice can get the treasure.

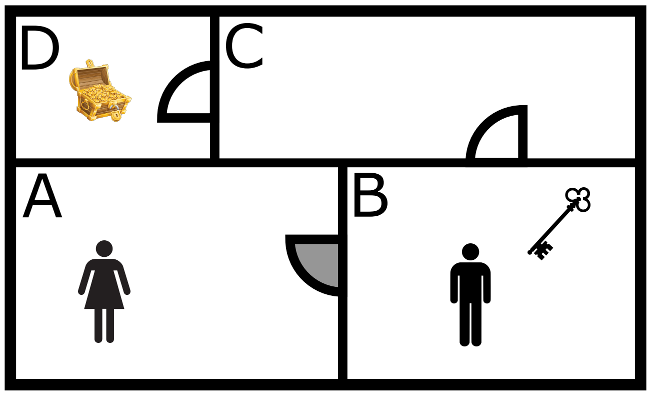

Figure 1: The illustration of the task. The grey door is locked; key from room B is needed to open it. Room D contains the treasure Alice wants to get.

4 Assignment

Your goal is to use situation calculus formalism to resolve the planning tasks using the first-order logic (FOL). The assignment is to complete the provided template in the language used in TPTP[1].

4.1 Grading

The assignment is worth 10 points. The first introductory task is worth 2 points (each conjecture is worth 0.125 points), and the main task is worth 8 points. For the main task, implementation of the actions' effect axioms is worth 3 points, and implementation of the actions' frame axioms is worth 5 points. In order to complete the assignment, you have to get at least 5 points.

The existence of proofs themselves helps to evaluate the solutions, but it is not directly graded. Do not try to reach them at any cost (e.g., using intentional mistakes that make Prover9 exit as expected).

4.2 Running the provers

You can use the online interface available at `http://www.tptp.org/cgi-bin/SystemOnTPTP`; note that the computational resources are limited and if you leave the assignment to the last minute, you can have problems with running tasks there.

[1]`http://www.tptp.org/`

4.3 Submission

Submit in the upload system[2]. You are supposed to upload a zip file (one zipped folder without any additional subfolders) containing the solution. The solution should consist of two files with the main and introductory tasks named as `main_assignment_<username>.txt` (main task) and `introductory_assignment_<username>.txt` (introductory minitask), where `<username>` is your username.

Modify the solution template only at the places where told otherwise your solution might be problematic to grade (and in the worst case won't be graded), especially do not remove **the block** (you'll know what is meant when you look into the template).

4.4 Deadline

The deadline is **Tuesday 28.4.2020 23:59 CEST**

4.5 Notes

- implement the required steps one by one and often check whether it does what you expect (by using custom conjectures)
- the sooner you start working on the assignment, the more time you'll have for potential questions

5 About the main task template

The template is provided in the file `main_assignment_template.txt`.

5.1 Available actions

5.1.1 move(A,X,Y,S)

This action describes the move of agent A from location X to a reachable location Y. The action start in state S and at that state the location Y must be reachable from location X and agent A has to be capable of moving.

The action does not influence the location of agents and objects not involved in the action. It does not change the abilities to move, to pick up objects or to be picked up of any agent or object. It also does not modify the door or that an agent has something.

5.1.2 pick_up(A,K,X,S)

The action is that the agent A picked up an object K at the location X in the state S. The agent A has to be capable of picking up an object in the state, both the agent A and the object K has to be at the same location, and the

[2]`https://cw.felk.cvut.cz/brute/`

object K can be picked up. The agent A has the object K at the end of the action.

The action does not influence the location of agents and objects not involved in the action; the object K no longer needs a location as it always is with the agent A. It does not change the abilities to move, to pick up objects or to be picked up of any agent or object. It also does not modify the door or that an agent has something.

5.2 open_door(D,A,R1,R2,K,S)

The agent A can open the door D leading from room $R1$ to room $R2$ using the key K in the state S. The key has to be able to open the door from location $R1$ and the agent has to be at that location. The agent has to have (`has(...)`) the key K and the door has to be closed.

The action does not influence the door state of doors not involved in the action. It does not change the abilities to move, to pick up objects, or to be picked up of any agent or object. It also does not modify any location or that an agent has something.

5.3 Available predicates

5.3.1 location(O,R,S)

An agent or an object O is in the room R in the state S.

5.3.2 door(D, R1, R2, DS, S)

This describes that there is a door named D leading from room $R1$ to room $R2$ and is in the door state DS in the state S. The available door states DS are `open` and `closed`.

The axiom named `bidirectional_doors` makes all the doors bidirectional, i.e., if a door leads from room $R1$ to room $R2$, then it also leads from room $R2$ to room $R1$.

5.3.3 reachable(X,Y,S)

It says that the location Y is reachable from location Y in the state S, i.e., there is a path of open doors connecting the locations X and Y in the state S.

5.3.4 can_move(A,S)

The agent A can move in the state S. Note that the agent can lose the ability to move (e.g., the agent can get too sick to move) as the ability is defined for individual states.

5.3.5 can_pick(A,S)

The agent A can pick up objects in the state S. Note that the agent can lose the ability to pick up an object (e.g., the agent can get too tired to pick up objects) as the ability is defined for individual states.

5.3.6 has(A, O, S)

An agent A has an object O in the state S.

5.3.7 can_open_lock(K,D, R1, R2)

It says that the key K can open the door D leading from the room $R1$ to the room $R2$. The key can open the door only from room $R1$. This ability is permanent because if a door has a lock that can be opened with a certain key, they cannot lose the lock.

5.3.8 can_be_picked(O)

The object O can be picked up, i.e., it is not a permanent component of unmovable parts such as the rooms or the doors.