KTH, DD2380 Artificial Intelligence

PDDL and SWISH

1 Transportation Task

In this exercise, we are going to look into encoding a planning domain using PDDL. Consider a fictional country that is suffering from a big pandemic. After months of research a medical team is successful in creating 2 different vaccines. However, one of them must be kept below a certain temperature at all times, therefore, its transportation requires a freezing unit. We are now concerned with the logistics associated with the distribution of these vaccines. There are 3 major cities and at least a warehouse to store the vaccines. Inside each city there is at least one hospital that is equipped to deliver any vaccine. To travel within a city, a short-range vehicle is needed while to travel between cities or between a city and a warehouse we need a long-range vehicle. While short-range vehicles are guaranteed to be able to transport temperature-sensible content, the long-range ones are not. In order to do so the vehicle must first load a freezing unit. It is possible for a vehicle to transport multiple boxes of vaccines.

The five basic actions to solve this problem are:

- **Load vaccine:** the vaccine is loaded into a vehicle;
- **Unload vaccine:** the vaccine is unloaded from a vehicle;
- **Load freezer:** the freezer is loaded into a vehicle;
- **Travel long:** Travel with a long range vehicle;
- **Travel short:** Travel with a short range vehicle.

1.1 Defining the Domain

Download the package **vaccine.zip** from Canvas. Your task is to complete the file **vaccine-domain.pddl** to formalize the planning domain. The syntax used in the file is a standardized syntax used in state-of-the-art PDDL solvers, such as in this on-line editor and solver [2]. There are numerous examples of problems encoded in this syntax under the Import tab in this tool. There are also numerous tutorials on this syntax, for instance this one [1]. The relevant tab to explore there is PDDL Background.

Note that the file vaccine-domain.pddl will only contain the definition of the *domain. The problem instance* including the definition of objects in the world, the initial state and the goal specification are given in a separate file. You can find one problem instance for the domain in this exercise in **problem-1.pddl**, illustrated in Figure 1.1).

In this scenario, the vaccines have already been delivered but there is a shortage in the hospital 3 (H3). We have some leftovers of vaccines at hospital 1 (H1) and we want to deliver them to hospital 3.

Your task is to complete only the code for each of the five actions, which involves writing the parameters, the precondition and the effect. Note that each action comes with a comment that gives more details that the brief domain introduction above. All the predicates you are allowed to use are already given in the file. You will not need to define any requirements or functions. Figure 1.1 shows the road network and initial state of the objects in the world.

You can use the above mentioned on-line editor and solver [2] to see whether your domain definition allows to find a solution to problem-1.pddl. It should one although it may not be most efficient (don't worry about that).

Please, submit your solution to Exercise 1.1. as a text file vaccine-domain.pddl in PDDL1.1 assignment. Make sure that your submitted file does not contain syntax errors.

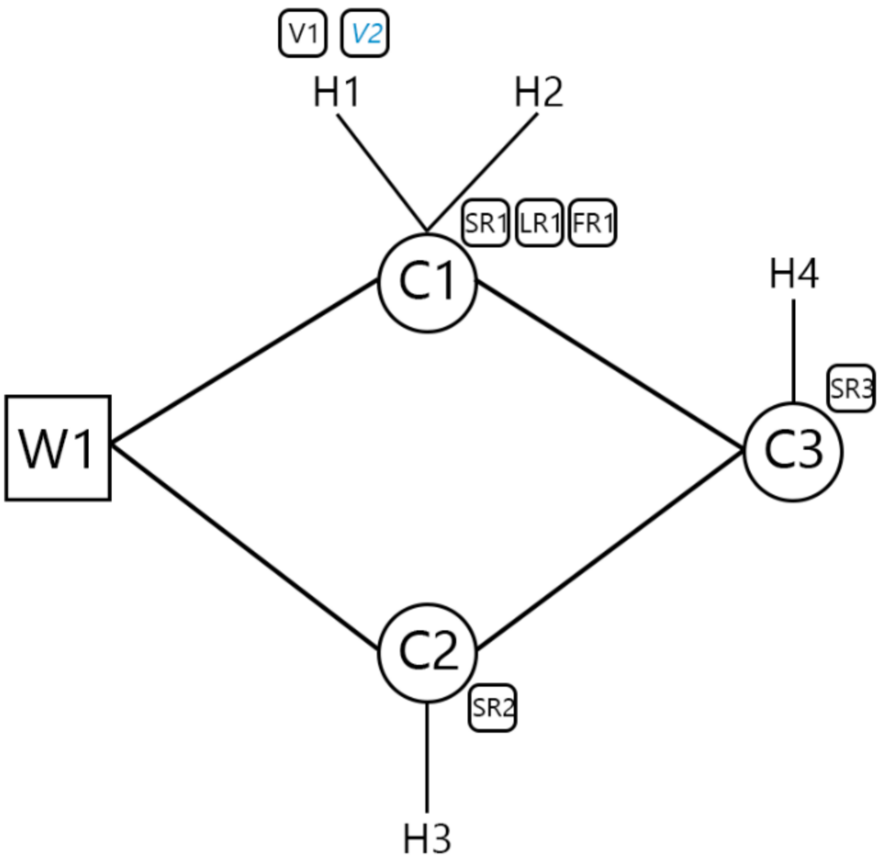

Figure 1.1: Transport network and initial conditions of the problem. V_i are the vaccines. In blue italics there are the temperature-sensible ones. LR_i and SR_i stand for long range and short range vehicle. FR_i is the freezing unit, C_i are the cities and W_i are the warehouses.

1.2 Designing a Problem

In the previous exercise you defined the planning domain and tested it in a given problem instance. Now the task is to adapt problem-1.pddl in order to reflect the objects and initial conditions defined in Figure 1.2.

The goal is to deliver one vaccine box to each hospital. Please, submit your solution to Exercise 1.2 as a text file problem.pddl in PDDL1.2 assignment.

References

[1] A PDDL 2.1 tutorial, `https://www.cs.cmu.edu/afs/cs/project/jair/pub/volume20/fox03a-html/JAIRpddl.html`, Accessed: 2018-09-26

[2] An online PDDL editor and solver, `http://editor.planning.domains/`, Accessed: 2021-02-22

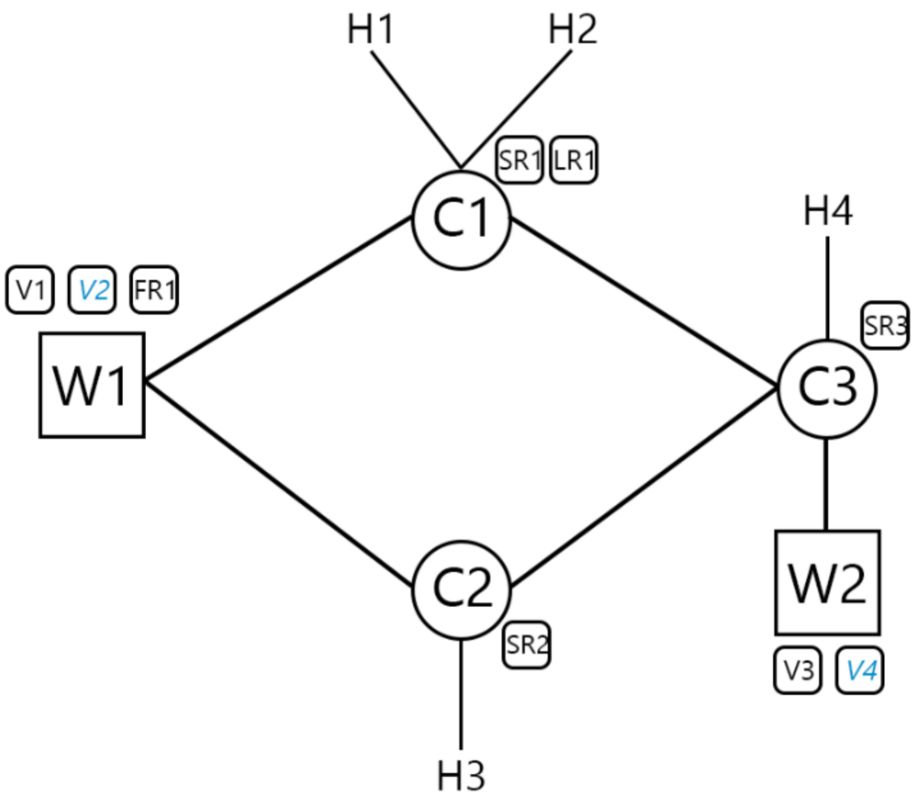

Figure 1.2: Transport network and initial conditions of the second problem to design. The goal is to deliver one vaccine box to each hospital.

2 SWISH

In this exercise we are going to solve a puzzle using Prolog. For that, you are going to use an online editor called SWISH [1]. The website also provides users with examples and tutorials; check, for instance, the Einstein's Riddle example.

For this exercise, let's suppose a group of 4 citizens (`c1`, `c2`, `c3` and `c4`). Each of them has a different age (`20`, `40`, `70` and `90`) and a different profession (`doctor`, `nurse`, `teacher` and `driver`). Each of them gets either `vac60` or `vac99` and goes to get vaccinated either to `karolinska`, `sodersjukhuset` or `sofiahemmet`.

We know that two citizens are in the same group if they receive the same vaccine and go to the same hospital if they get vaccinated there.

- F1 `c1` is a `doctor`.
- F2 `c3` is `20` years old.
- F3 `c2` is a `teacher`.
- F4 A citizen goes to `karolinska` to get `vac99` but not `c4`.
- F5 The `teacher` gets vaccinated in `sofiahemmet`.
- F6 The `nurse` gets vaccinated in `karolinska`.
- F7 The citizen aged `70` goes to the same hospital as `c3`.
- F8 A citizen gets `vac60` in `sodersjukhuset`.
- F9 The citizen aged `40` is not in the same group as `c2`.
- F10 The driver goes to the same hospital as `c4`.
- F11 The citizens aged `70` and `90` are in the same group.

Download the file **SWISH.txt** from Canvas and paste its contents in the online editor [1]. The skeleton of the problem is ready. Your task is to fill in the missing facts.

How to submit: Under the assignment SWISH, upload SWISH.txt. Make sure that your submitted file does not contain syntax errors.

References

[1] An online Prolog editor and solver, `https://swish.swi-prolog.org/`