UNIVERSITAT POLITÈCNICA DE VALÈNCIA

DSIC DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Escuela Técnica Superior de Ingeniería Informática

etsinf

Dpto. Sistemas Informáticos y Computación
Escuela Técnica Superior de Ingeniería Informática
UNIVERSITAT POLITÈCNICA DE VALÈNCIA

INTELLIGENT SYSTEMS

LAB ASSIGNMENT 1

DESIGN, IMPLEMENTATION AND EVALUATION OF A RULE-BASED SYSTEM

Airport terminal

September 2019

Airport terminal

The problem we want to solve consists in moving luggage between locations of an airport terminal by using an airport luggage car. The airport layout is shown in Figure 1:

- CHECK-IN: this is the location in which passengers of departure flights drop their luggage.
- BAGGAGE CLAIM: this is the location in which passengers of arrival flights pick up their luggage
- GATE 1 to GATE 8: locations for planes that are going to take off (departure flights) and for landed planes (arrival flights).

The luggage of a departure flight is transported from the CHECK-IN point to the gate where the corresponding plane is located. The luggage from an arrival flight is transported from the gate at which the plane stalls (lands) to the BAGGAGE CLAIM point.

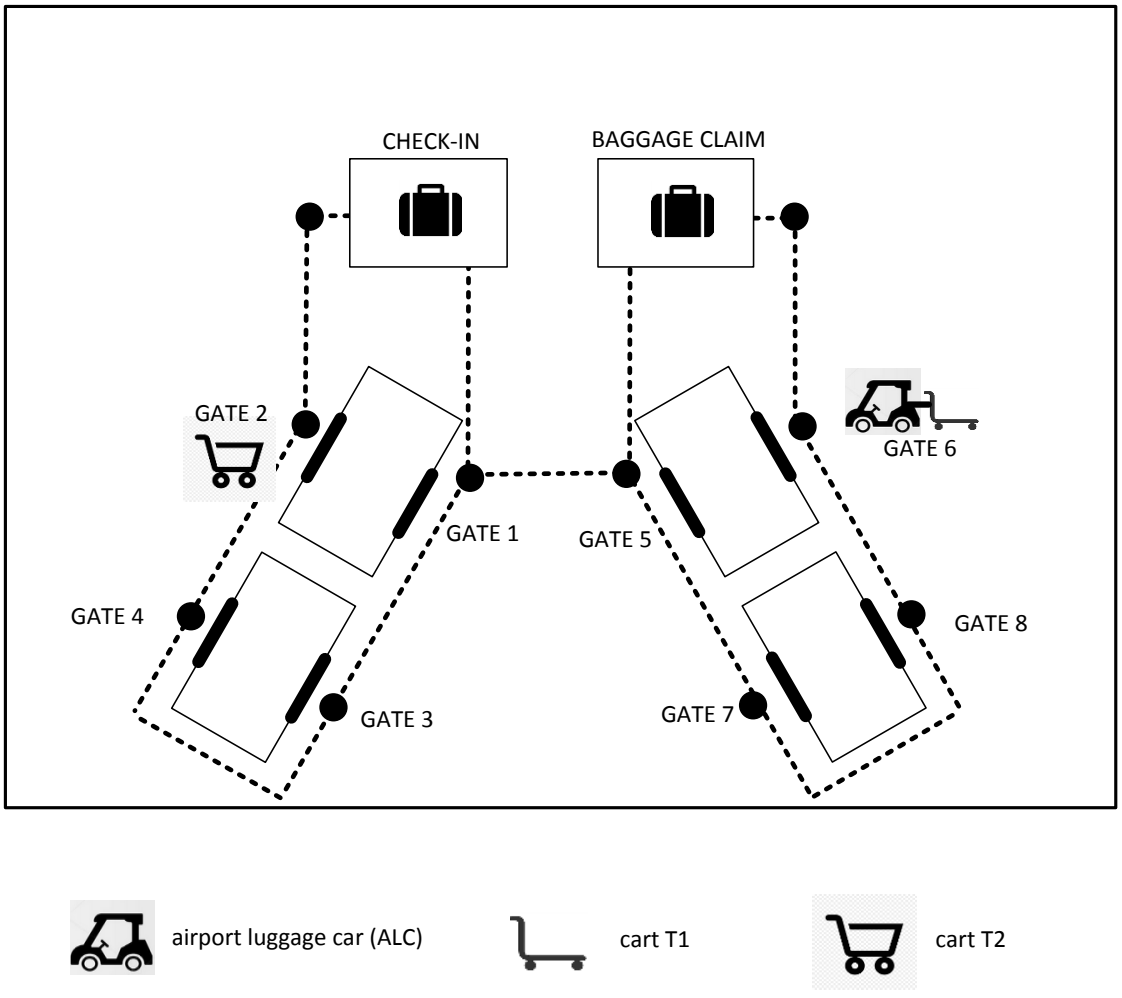

Figure 1. Terminal airport

There is **only one** airport luggage car (ALC) and **various** carts for transporting suitcases between locations (see the icons at the bottom of Figure 1). The ALC and the carts are initially located at a point of the airport. There are two types of carts:

- cart type 1 (T1): it can only carry suitcases which weight is at most 15 kg (0 to 15 Kg.)
- cart type 2 (T2): it can only carry suitcases of more than 15 kg (16 to 23 Kg.)

Every suitcase is labeled with a sticker that shows its weight. Carts can transport an unlimited number of suitcases.

Carts are hooked to the ALC and unhooked from it. Figure 1 shows a cart of type T1 hooked to the ALC at GATE 6. Due to the characteristics of the ALC, only one cart can be hooked at a time. For example, in Figure 1, if we want to transport a suitcase of more than 15 kg., the currently hooked cart of type T1 must be unhooked and then hitch a cart of type T2 to the ALC.

The ALC can only move through the dotted lines shown in Figure 1. For example, to go from the CHECK-IN to GATE 7, the ALC can follow the path that goes by GATE 1 and GATE 5 to finally reach GATE 7, which would be the shortest path. Another possibility to reach GATE 7 is heading for GATE 2 and follow all the way through GATE 4, GATE 3, GATE 1, GATE 5 and GATE 7.

An example of initial situation is the following:

- the ALC is at GATE 6 (see Figure 1)
- a cart of type T1 is hooked to the ALC (see Figure 1)
- there is a cart of type T2 located at GATE 2 (see Figure 1)
- there are four suitcases to move:
    - S1 is at the CHECK-IN point, weighs 12 kg., and its destination is GATE 3.
    - S2 is at the CHECK-IN point, weighs 18 kg., and its destination is GATE 5.
    - S3 is at GATE 1, weighs 20 kg., and its destination is the BAGGAGE CLAIM point.
    - S4 is at GATE 6, weighs 14 kg., and its destination is the BAGGAGE CLAIM point.

The work to do consists in solving this airport terminal scenario, using a state-based representation of the problem, with a RBS implemented in CLIPS. The BREADTH and DEPTH search strategies will be used to run the CLIPS program and solve the problem:

1. The program will request the user the maximum depth level of the tree to develop (see example of the 8-puzzle problem).
2. The program will return the depth level and number of generated nodes of the solution found for each execution of the RBS with the search strategies (no need to return the solution path).
3. Use a generic knowledge representation that is readily modifiable. The program should allow to define a variable number of suitcases (weigh, origin and destination) as well as a variable number of carts of both type T1 and T2 without having to modify the rules. We will assume for all problem instances that only one ALC exists and that **at most** one cart can be hooked to the ALC at a time, being able to switch the cart from one move to another.

4. All the initial information can be directly written within the `deffacts` command.

GUIDELINES

1. The pattern that will represent the information of a problem state should only comprise the dynamic information that is modifiable from one state to another, like:

    - the suitcases' location
    - the location of the ALC
    - the position of the carts; carts can be hooked to the ALC or not – in case a cart is not hooked to the ALC, it will be located at a point of the airport (e.g. the cart located at GATE 2 in Figure 1)

2. We can distinguish two types of static information in the problem (information that is unchangeable along the execution of the RBS):

    - Static information for all the problem instances like the airport layout, the paths connecting the airport locations or the two types of carts.
    - Static information of one given problem instance such as the weight, origin and destination of the suitcases, which is all defined in the initial state of the problem instance.

3. The applicable actions in this problem are: (1) loading a suitcase in the cart at its original location; (2) unloading a suitcase from the cart at its destination (for loading/unloading luggage, the cart must be hooked to the ALC); (3) hooking a cart to the ALC (the cart must be empty); (4) unhitching the cart from the ALC (the cart must be empty); and (5) moving the ALC.

EVALUATION OF THE LAB ASSIGNMENT

The assessment of the lab assignment will be through an **INDIVIDUAL** test that will take place on Wednesday October 16. Two turns, each of 45 minutes, will be arranged on October 24: first turn will be from 8.00 to 8.45 and second turn from 8.45 to 9.30.

The test will include some questions about the implementation, evaluation of the RBS or modifications of the code to address a new functionality in the Sokoban game.

A task in PoliformaT will be created for the lab assessment. The CLIPS program that solves the Sokoban game described above must be uploaded to the PoliformaT task (CLIPS program before the exam) as well as the new CLIPS code that tackles the modifications requested in the exam (CLIPS program after the exam) (**IMPORTANT**: indicate clearly the name of the two group members -if this is case- in the CLIPS file that contains the program that solves the problem presented in this document).