AUTONOMOUS SOFTWARE AGENTS
ASSIGNEMENT A1 - NATIONAL VACCINE DISTRIBUTION

Francesco Ferrini - 224010

DIRECTORY TREE

```
- Assignement
  - Single_agent
    - big_problem
        - vaccines-problem-big.pddl
    - small_problem
        - vaccines-problem-small.pddl
        - vaccines-domain-small.pddl
        - plan
  - Multi_agent
    - big_problem
        - problem-MA-big.pddl
        - domain-MA-big.pddl
        - maplan-run.sh
    - small_problem
        - problem-MA-small.pddl
        - domain-MA-small.pddl
        - maplan-run
        - plan
  - ASA-ASSIGNEMENT A1
  - README
```

SINGLE AGENT

- INTRODUCTION

There is a central point (light purple circle) in Italy that receives all the vaccine boxes, and from there the vaccine boxes shall be distributed to one designated central location (red circle), one for each region. Once the vaccine boxes reach the designated central locations in every region, such vaccine boxes shall be distributed to the different provinces (dark green circles) of the region. From the designated location in each province the vaccines shall be distributed to the different health districts (light green circles) within the same region. Transportation of vaccine boxes is performed with different transport agents (planes, trucks, and drones). Initially all the vaccine boxes are in the central location of the state. The goal is that there is at least one vaccine box in each health district.

- CONTRAINTS SATISFACTION

There are 20 regions named as in the real Italy scenario. Each region has 6 provinces (a part EMILIAROMAGNA region that has 7 provinces) named with the acronym of each real province (since in Italy in some regions there are less than 6 provinces, some of them are invented). Each province has 3 health districts named with the province name adding 1, 2 and 3 to the name. As in the real scenario, I decided that airports can only in provinces. There are regions in which all provinces has no airports. Between regions and between regions and provinces, the trucks can move. Between a province and an health district a drone can move. Between 2 provinces that have airport, an airport can move.
Each transport can load and unload vaccine boxes and its capacity is limited. In particular drones have a capacity of one, the planes have a capacity of 10 and the trucks have a capacity of 5. In this scenario there is 1 drone for each province (a total of 121 drones) and there are 10 planes and 20 trucks.

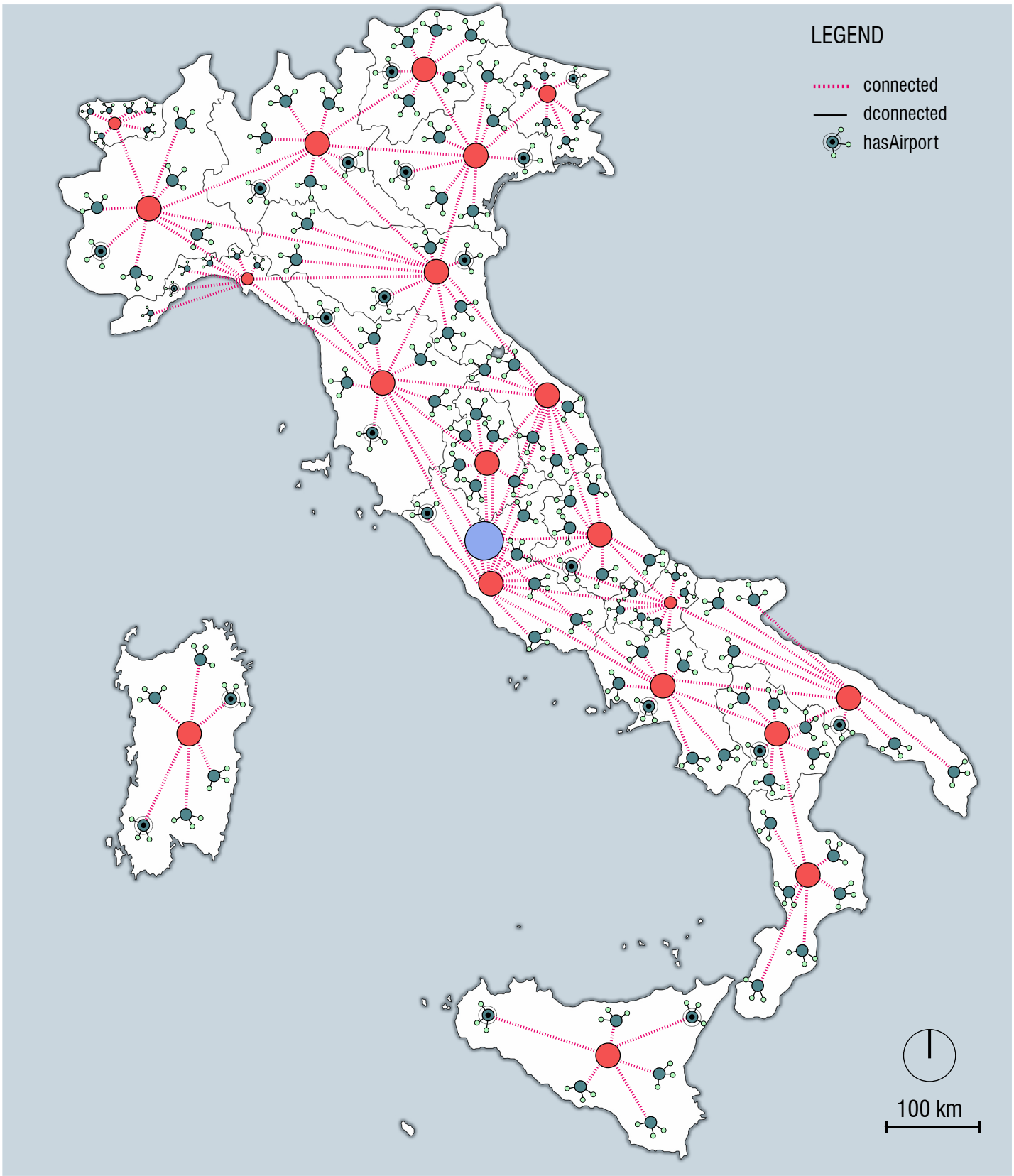
- DOMAIN FILE

In the domain file we find:

A. REQUIREMENTS
   For this assignment I used strips and numeric-fluent

B. PREDICATES
   Here there are the predicates for the transport (`plane`, `drone`, `truck`) and for the vaccine boxes (`box`). Then there is a predicate in order to check if a drone is empty or not since it has a capacity of 1 and so can be empty or full (`emptyDrone`). In my project central point, regions, provinces and health districts are considered all locations with the `location` predicate. Then there are 2 types of connections between locations; in particular we find the predicate `connected` (that connects a region with neighbor regions and regions with their own provinces) and the predicate `dconnected` (that connects provinces with their own health districts). These types of connections are shown in Fig. 1. Then there is a predicate `hasAirport` used to describe whether a province has an airport. The predicate `at` is used to describe in which location is a transport or a box and `in` is used to describe in which transport is a vaccine box after the loading. For last we have the predicate for the `capacity` of the transports.

C. FUNCTIONS
   As functions we have only the capacity one.


LEGEND
connected
dconnected
hasAirport
100 km

D. ACTIONS

In my domain file there are 3 actions for each transport. The `fly_plane` action describe the moving action that a plane can do. In particular if the plane is at a dep location, this location has an airport and also the dst location has an airport, the plane can move at dst location. Similar is the `move_truck` action used for the trucks. The difference is that the truck can move between 2 locations if and only if they are connected with the connected predicate. Then in order to move the drones there is the `fly_drone` action that is used to move the drones between 2 locations that are connected with the dconnected predicate.

For loading the transports we have 3 actions: `load_plane`, `load_truck` and `load_drone`. They are similar but the first two uses the capacity function, so each time a plane or a truck load a vaccine box, the capacity increases of one. In order to limit the capacity of planes and trucks in the preconditions of the `load_plane` action we have that the capacity must be smaller than 10 and in the `load_truck` action we have that the capacity must be smaller than 5. For the drones instead, since they can load maximum one box, it's used the `emptyDrone` predicate; so a drone can load a box only if the drone is empty.

Then we have the 3 actions for the unloading. They are similar to the loading actions. So the `unload_plane` and the `unload_truck` actions checks whether the capacity is greater than 0 (there is some boxes into the plane or into the truck) and in this case they can be applied. In this case the box is left at the location where the transport is and the capacity of the transport decreases of 1. Instead the `unload_drone` action is used for the drones. It can be applied only if the drone is not empty and in this case the box can be left at the location where is the drone (and the drone becomes empty).

- PROBLEM FILE

In the problem file we find:

A. OBJECTS

Here are added all the objects we need. In particular there are the names for the various locations, the names for the drones, trucks, planes and boxes and the name for the central point. In order to better understand the structure, in Fig.2 is an example. Then are instantiated the names for the planes (plane1, …, plane10), for the trucks (truck1, … , truck20) and the name for the central point (CENTRAL_VACC_POINT).

```
LAZIO
      RM        drone1
        RM1     box1
        RM2     box2
        RM3     box3
      VT        drone2
        VT1     box4
        VT2     box5
        VT3     box6
      LZ1       drone3
        LZ11    box7
        LZ12    box8
        LZ13    box9
      RI        drone4
        RI1     box10
        RI2     box11
        RI3     box12
      LT        drone5
        LT1     box13
        LT2     box14
        LT3     box15
      FR        drone6
        FR1     box16
        FR2     box17
        FR3     box18
```

Fig. 2
Here is the configuration for the LAZIO region. It has 6 provinces: RM, VT, LZ1, RI, LT and FR and each province has 3 health districts (for RM the 3 health districts are RM1, RM2 and RM3). Then for these 6 provinces are instantiated 6 names for the drones (drone1, …, drone6) and since there are 18 health districts, are instantiated 18 names for the vaccine boxes (box1, …, box18).
The configuration is the same for all the regions.

B. INIT

In the init configuration are instantiated the vaccine boxes using the box `predicate`, the planes with the `plane` predicate, the truck using the `truck` predicate and the drones with the `drone` predicate.

Then all the locations (with the `location` predicate) are instantiated. I decided to use only the `location` predicate for central point, regions, provinces and health districts.

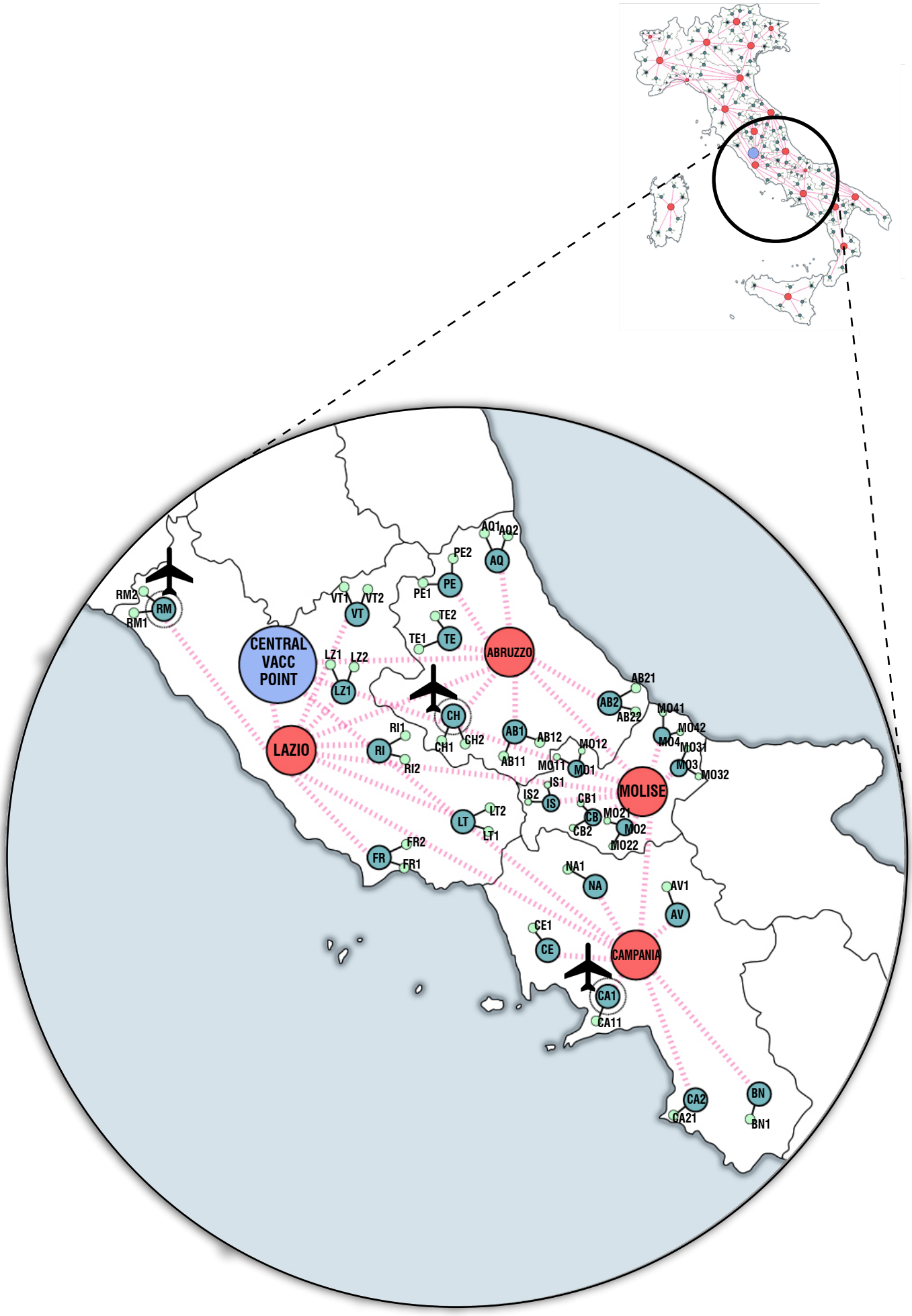
Then with the predicate `at` I decided, as asked in the constraints, that al boxes are in the CENTRAL_VACC_POINT initially.

With the capacity function I set to 0 the capacities for planes and trucks and using the emptyDrone predicate I set as empty all the drones.

Then we find the connections between locations. Since I decided that the trucks can move only between regions (that are neighbors in the real scenario) and between a region and its own provinces, the connection for the trucks movement is set with the predicate `connected` (For example, watching at the Fig. 3 in the enlargement is clear that LAZIO is connected with CAMPANIA, ABRUZZO and MOLISE and the same connection is used for LAZIO with RM, LAZIO with VT, LAZIO with LZ1, LAZIO with RI, LAZIO with LT and LAZIO with FR). Instead, since only drones can take boxes from a province and bring it to the health districts in that province, the connections between provinces and their health districts are set with the predicate `dconnected` (For example, watching at the Fig. 3 in the enlargement is clear that RM is dconnected with RM1, RM2).

Then we find the declaration of the provinces with the airports using the predicate `hasAirport`.

In the end of the init configuration all the vehicles are putted in some location in the map (all the vaccine boxes are initially in the CENTRAL_VACC_POINT).


CENTRAL VACC POINT
LAZIO
ABRUZZO
MOLISE
CAMPANIA
RM
RM1
RM2
VT
VT1
VT2
RI
RI1
RI2
FR
FR1
FR2
CE
CE1
CA1
CA11
CA2
CA21
AV
AV1
BN
BN1
NA
NA1
LT
LT1
LT2
CH
CH1
CH2
TE
TE1
TE2
PE
PE1
PE2
AQ
AQ1
AQ2
LZ1
LZ2
IS
IS1
IS2
CB
CB1
CB2
AB1
AB11
AB12
AB2
AB21
AB22
MO1
MO11
MO12
MO2
MO21
MO22
MO3
MO31
MO32
MO4
MO41
MO42

C. GOAL

In the goal it is specified where each box must be located at the end using the (`at ?box ?health_district`) predicate.

- RESULTS

Since the whole problem has a very big complexity and the Metric ff planner is not able to handle a such big tree, I decided to reduce the problem using only 4 regions, 6 provinces for region and a total of 42 health districts as is shown in the enlargement of the Fig.3 . Using this configuration for the problem file, the metric ff planner is able to find a solution in the time reported in Fig. 4.

```
time spent:    0.03 seconds instantiating 205167 easy, 0 hard action templates
               0.02 seconds reachability analysis, yielding 4678 facts and 18912 actions
               0.01 seconds creating final representation with 4651 relevant facts, 20 relevant fluents
               0.08 seconds computing LNF
               0.04 seconds building connectivity graph
            1401.78 seconds searching, evaluating 106799 states, to a max depth of 4
            1401.96 seconds total time
```

Fig. 4
Computational time to find a solution

MULTI AGENT

- INTRODUCTION

The problem is the same but in this case it has to be modeled with a multi agent architecture.

- DOMAIN FILE

In the domain file we find:

A. REQUIREMENTS

For this assignment I used strips, typing (to create types), multi-agent and numeric-fluent.

B. TYPES

In this section we specify the different types; in this case there is the type `object` that comprehends `agent` and `location`, and the type `agent` that includes `truck`, `drone` and `plane`.

C. PREDICATES

Here we have the predicate `connected` (which connects 2 types location), the predicate `at` in order to specify that an agent is as a particular location; then the predicate `hasAirport` to describe which province has an airport and for last we have 2 predicates, `is-region` and `is-hd` to specify whether a location is a region or an health district.

D. FUNCTIONS

Here is used only the function `capacity` that specify the number of vaccine boxes that there are in a location or that are in a particular agent.

E. ACTIONS

There are 3 functions for move the agents. In particular `drive_truck` is the action that allows a truck to move between 2 locations only if the destination is not an health district. `Fly_drone` action allows the drone to move between 2 locations only if the destination location is not a region. Instead `fly_plane` action is used to move a plane between 2 locations that have an airport.

The functions for the loading and unloading are the same for all the 3 agents but the only difference is the maximum capacity for the 3 because drones have a maximum capacity of 1, planes have a maximum capacity of 20 and trucks have a maximum capacity of 10. Each time an agent loads, the capacity of the location decreases of and his capacity increases of 1. At the opposite during the unloading the capacity of the location increase of 1 and the capacity of the agent decreases of 1.

- PROBLEM FILE

In the domain file we find:

A. OBJECTS
   In the object part are defined all the object types we need so 20 trucks, 10 planes, 121 drones and all the locations' names as in the single agent problem.

B. INIT
   In the init section, firstly are defined all the capacities. Since we want that each health district has 1 vaccine box (capacity = 1), we initially set that the capacity for all the locations are 0 but for the CENTRAL_VACC_POINT that has a capacity of 363 (vaccine boxes). Then are set to 0 also the capacities for the different agents.
   Then are set all the connections between locations with the predicate `connected`.
   Since the agents (trucks and drones) need to know if a location is a region or an health district, we set the regions with the predicate `is-region` and the health district with the predicate `is-hd`.
   At the end of the init, all the agents are placed in some location.

C. GOAL
   As goal, all the health districts at the end must have a capacity equal to 1, that means they all have a vaccine box.

D. FORMULATION PROBLEMS
   Since the maplan planner doesn't support numeric fluent (and neither conditional effects), this formulation of the problem can't be tried on that planner.
   For this reason I decided to develop a smaller problem also for the multi agent scenario in order to work with a multi agent problem and try a solution.
   In particular in this case I considered 2 regions, 4 provinces and a total of 4 health districts. Then there are 2 trucks, 4 drones and (since it was not possible to use fluents) 4 boxes that in the domain file are considered as objects.
   In the domain file so we don't use the capacity and so there are no constraints on it for the trucks. For the drones I used another predicate in order to check whether it is empty or not (`empty`).
   As goal we have that each box must be at a different health district using the predicate `at`.

E. RESULTS
   The results for this methodology is a plan that can be found in the multi agent small problem directory.