**Arianna Bettin – K1763730, Hani Tawil - K1533307**

The plan scenario annotated below, characterizes a model of the upcoming drone delivery amazon is trying to implement for the near future. In this model we govern the logistics of how the drone picks up and delivers (by flight) the package whilst also sustaining a healthy amount of charge to function. We specialize the model further to also take into account the transportation of goods from the warehouse in which the products are stored to the drone-friendly post offices.

The model provides a good foundation to emphasize the importance of explainable AI by applying many simple constraints which become more and more complex as the problem world grows in size. These constraints include:

* Limited size available in trucks
* Charge management of drones
* Problems where there are more drones than charging bases
* Specific routes that only certain vehicles can take
* A segregation of the map so that certain post offices are more efficient to fly to certain locations.

With these questions we can ask the planner why it choose to make specific moves to try and optimize the world around the constraints listed above.

The types identified and defined in the domain are as follows:

* A generic type, **"object",** and its instances, "**place"**, and "**locatable",** instances of which can be associated with a "location" type.
* The **“drone”** (instance of **“vehicle” and “locatable”**) – The machine which will transport packages from the post office to the destination.
* The **“truck”** (instance of **“vehicle” and “locatable”**) – Vehicle which will transport goods only from the warehouse to either post office.
* The **“item”** (instance of **“locatable”**) – The object the drone will carry to its destination
* The **“vehicle”** (instance of **“locatable”**) – The object which encapsulates the drone and truck
* **“droneBase”** is also given a type so that we are able to distinguish between the drop-off locations and the locations in which the drone need to return to and recharge at.

The predicates identified and defined in the domain are:

* **“(at ?item - locatable ?loc - place)”** – this helps represent the locations of a specific object
* **“(available ?d - drone)”** – this shows if a drone is available for transport
* **“(charging ?d – drone)”** - to show that the drone is charging
* **“(in ?item – locatable ?v – vehicle)”** – checks to see what package a specific drone or truck is carrying
* **“(air-link ?x ?y - place)”** – used to recognize a path which only a drone is able to take.
* **“(link ?x ?y - place)”** – used to recognize a path which only a truck is able to take.
* **“(droneBaseAvailable ?db - droneBase)”** – True when no drone is using the base for charging since only one drone can be charged at a time.

The functions identified and defined in the domain are as follows:

* **“(charge-level ?d – drone)”** – the percentage of charge left in the drone
* **“(charge-required ?x ? y – place**)” – the amount of charge required to reach a location.
* **“(flight-time ?x ?y – place)”** – the time it takes to move from one location to another via drone
* **“(drive-time ?x ?y – place)”** – the time it takes to move from one location to another via truck
* **“(loaded-packages ?t - truck)”** – used to track the amount of packages a truck is currently holding

The durative actions that are defined in the domains are:

* **“ATTACH-PACKAGE”** – with “item”, “place” and “drone” as the parameters and the duration of 10 units. To accomplish this action the drone needs to available, both the drone and the package need to be at the same location. This action will make the drone occupied and change the location of the package to the drones location.
* **“RELEASE PACKAGE”** – with the same parameters as “attach package” but with opposite effects.
* **“FLY-DRONES”** - with "drone” and two "location" as parameters. The duration is equal to the "flight-time" function associated with the two "location" points. In short, at the end of the duration, the "location" of the "drone" will change to that of the target "location". The drone needs to make sure that it has enough charge to get to the location and back. It will not take the trip if it cannot make it.
* **FLY-DRONEBASE**” – With the same parameters as “fly-drone” but instead also includes the dronebase object to make sure its going back to the right location as well as calculating the charge required correctly.
* **“RECHARGE-DRONE”** – with “drone”, “place” and “base” as parameters. The duration is equal to 100 minus “charge-level ?d”, therefore the charge rate is 1% for every unit of time. The “drone” must be at the “droneBase” to be recharged and at the end of the duration its charge level will be equal to 100%.
* **“LOAD-TRUCK”** – with “item”, “place” and “truck” as parameters. The duration is equal to 10 units. The “truck” and “item” need to be in the same location and the truck must not have more than 10 packages in it. By the end of the action the item is in the truck and the load of the truck has increased by one
* **“UNLOAD-TRUCK”** – with the same parameters as “load truck” but has the opposite effects.

PDDL Domain file.

(define (domain Delivery)

(:requirements :strips :typing :time :numeric-fluents :durative-actions :conditional-effects)

(:types

place locatable - object

vehicle truck drone item dronebase - locatable

drone truck - vehicle

)

(:predicates

(at ?item - locatable ?loc - place)

(in ?item - locatable ?v - vehicle)

(available ?d - drone)

(link ?x ?y - place)

(air-link ?x ?y - place)

(charging ?d - drone)

(drone-base-available ?db - dronebase)

)

(:functions

(drive-time ?x ?y - place)

(flight-time ?x ?y - place)

(charge-required ?x ?y - place)

(charge-level ?d - drone)

(loaded-packages ?t - truck)

)

(:durative-action LOAD-TRUCK

:parameters

(?item - item ?truck - truck ?loc - place)

:duration (= ?duration 10)

:condition

(and (over all (at ?truck ?loc))

(at start (at ?item ?loc))

(at start (< (loaded-packages ?truck) 10)))

:effect

(and (at start (not (at ?item ?loc)))

(at end (in ?item ?truck))

(at end (increase (loaded-packages ?truck) 1)))

)

(:durative-action UNLOAD-TRUCK

:parameters

(?item - item ?truck - truck ?loc - place)

:duration (= ?duration 10)

:condition

(and (over all (at ?truck ?loc))

(at start (in ?item ?truck))

(at start (> (loaded-packages ?truck) 0)))

:effect

(and (at start (not (in ?item ?truck)))

(at end (at ?item ?loc))

(at end (decrease (loaded-packages ?truck) 1)))

)

(:durative-action ATTACH-PACKAGE

:parameters

(?item - item ?drone - drone ?loc - place)

:duration (= ?duration 10)

:condition

(and (over all (at ?drone ?loc))

(at start (at ?item ?loc))

(at start (available ?drone)))

:effect

(and (at start (not (at ?item ?loc)))

(at end (in ?item ?drone))

(at start (not (available ?drone))))

)

(:durative-action RELEASE-PACKAGE

:parameters

(?item - item ?drone - drone ?loc - place)

:duration (= ?duration 10)

:condition

(and (over all (at ?drone ?loc))

(at start (in ?item ?drone)))

:effect

(and (at start (not (in ?item ?drone)))

(at end (at ?item ?loc))

(at end (available ?drone)))

)

(:durative-action DRIVE-TRUCK

:parameters

(?truck - truck ?loc-from - place ?loc-to - place)

:duration (= ?duration (drive-time ?loc-from ?loc-to))

:condition

(and (at start (at ?truck ?loc-from))

(over all (link ?loc-from ?loc-to)))

:effect

(and (at start (not (at ?truck ?loc-from)))

(at end (at ?truck ?loc-to)))

)

(:durative-action FLY-DRONE

:parameters

(?d - drone ?loc-from ?loc-to - place)

:duration (= ?duration (flight-time ?loc-from ?loc-to))

:condition

(and (at start (at ?d ?loc-from))

(over all (air-link ?loc-from ?loc-to))

(at start (> (charge-level ?d) (\*2 (charge-required ?loc-from ?loc-to)))))

:effect

(and (at start (not (at ?d ?loc-from)))

(at end (at ?d ?loc-to)))

)

(:durative-action FLY-DRONEBASE

:parameters

(?d - drone ?loc-from ?loc-to - place ?db - dronebase)

:duration (= ?duration (flight-time ?loc-from ?loc-to))

:condition

(and (at start (at ?d ?loc-from))

(over all (at ?db ?loc-to))

(over all (air-link ?loc-from ?loc-to))

(at start (> (charge-level ?d) (charge-required ?loc-from ?loc-to))))

:effect

(and (at start (not (at ?d ?loc-from)))

(at end (at ?d ?loc-to)))

)

(:durative-action RECHARGE-DRONE

:parameters

(?d - drone ?l - place ?db - droneBase)

:duration (= ?duration (- 100 (charge-level ?d))

:precondition

(and (at start (< (charge-level ?d) 100))

(at start (available ?d))

(at start (drone-base-available ?db))

(over all (at ?d ?l))

(over all (at ?db ?l)))

:effect

(and (at start (not (available ?d)))

(at start (not (drone-base-available ?db)))

(at start (charging ?d))

(at end (= (charge-level ?d) 100)) \*\*NOT SURE ABOUT THIS LINE (also no loadedpackages in problems files)

(at end (available ?d))

(at end (drone-base-available ?db))

(at end (not (charging ?d))))))