**The overall Problem**

A number of swarms of drones fly over a field and search for pests. When a swarm has spotted a pest it can deploy one of its drones to eliminate the pest. That action also makes that drone unusable. The pests are from different types. Eliminating some is more important than eliminating others and some aren’t pests at all but animals that should not be harmed. In this specific version the field is not necessarily a rectangular grid. Also, when a swarm flies over an area, the probability to find a pest there decreases to 0. In addition to that, the initial probability to find a pest in an area is a given constant.

**Requirements**

The requirements for the domain are: **concurrent, integer-valued, reward-deterministic, multivalued** and **constrained-state.** These mean that multiple actions can be taken at once; there are integer variables in the domain; calculating the reward doesn’t require any stochastic functions; there is an enumerated variable in the domain; there are action requirements. The reasoning behind each one will be seen further.

**Types**

There are 3 different types in the domain. The type swarm contains objects since those are our agents defined in each instance. Same goes for the type location. Each object of this type represents an area in which a pest or a swarm may be found. The type pest is enumerated and contains values: @low\_level, @mid\_level, @high\_level, and @animal. The reason for this is that we don’t have each pest as a single unit but we only work with probabilities of finding pests of each level.

**Non-fluents**

* **real rew\_el (?p)** is a reward for every eliminated pest of type ?p.
* **real rew\_ex**  is a reward for exploring an area that has not been previously visited by a swarm.
* **bool neighbours(?l, ?l1) whether** it is possible for a swarm to travel between ?l1 and ?l **.**

**State-fluents**

* **bool p\_found (?p, ?l)** determines if a pest-type ?p is found on location ?l. Can be true only if a swarm is in the area ?l and only with a specific probability.
* **int drones\_in(?s)** is the amount of drones in swarm ?s. Gets lower when a swarm eliminates a target. Can’t be less than 0.
* **bool swarm\_at(?s, ?l)** determines if a swarm ?s is on the area ?l.
* **real prob(?p, ?l) represents** the probability of finding pest ?p on area ?l (assuming there is a swarm at ?l).

**Action-fluents**

* **bool move (?s, ?l)**  this action moves swarm ?s from where it has previously been to ?l. Can only be done if its original placement neighbours the new one.
* **bool eliminate\_pest(?s)** this action eliminates a target in an area that ?s is in, which makes the target disappear and gives the appropriate reward.

**CPFs**

* **drones\_in'(?s) = drones\_in(?s) - 1 \* eliminate\_pest(?s);**

In other words, the number of drones in a swarm stays as it was previously except it decreases by 1 if “eliminate” action is taken.

* **p\_found'(?p, ?l) = [exists\_{?s : swarm} swarm\_at(?s, ?l)] ^ Bernoulli( prob(?p, ?l) ));**

That means that a pest can be found if and only if there is a swarm on the same location and only with probability (prob(?p, ?l)).

* **swarm\_at'(?s, ?l) = [ drones\_in(?s) > 0 ] ^ [ forall\_{?l1 : location} (move(?s, ?l1) => (?l1 == ?l)) ] ^ [ (forall\_{?l1 : location} ~move(?s, ?l1)) => swarm\_at(?s, ?l)] ;**

The last CPF is equivalent to saying that whether a swarm is in an area ?l depends on three factors:

1. Number of drones in the swarm must not be 0
2. If the swarm moves anywhere, it only moves to ?l
3. If the swarm doesn’t move then it can be only be in ?l only it was in ?l previously.

**Reward**

**reward = sum\_{ ?p : pest, ?l : location}[ rew\_el(?p) \* ( p\_found( ?p, ?l ) ^ exists\_{s:swarm}[swarm\_at(?s, ?l) ^ eliminate\_pest(?s)]) +rew\_ex \* (( prob(?p, ?l)~=0 ) ^ exists\_{?s : swarm}[swarm\_at(?s, ?l)]) ];**

In order to get the reward we sum over every location and pest, 2 things:

Reward for elimination: requires for a pest to be found, a swarm being in an area, and that swarm eliminating the pest.

Reward for exploration: we can say that an area is explored when it has a pest with a probability of finding it different from zero and a swarm on the same area (a swarm is on the location but the probability wasn’t updated yet to 0, since it requires one full turn).

**State-action- constraints**

* **forall\_{ ?s : swarm, ?l : location, ?l1 : location } [ move( ?s, ?l ) ^ move( ?s, ?l1 ) => (?l == ?l1)];**

That is a very classy way of saying that there is only 1 “move” action that is taken by a swarm.

* **forall\_{?s : swarm, ?l : location, ?l1: location } swarm\_at( ?s, ?l1 ) ^ [~neighbours(?l, ?l1)] => ~move(?s, ?l);**

In other words: for every swarm and 2 areas, if the swarm is at one of them and those two are not neighbouring areas then the swarm can’t move to the other one. Which means that swarms can move only between neighbours.

* **forall\_{?s: swarm} [drones\_in(?s) = 0 => ~eliminate\_pest(?s)];**

The last constraint says that a swarm can't eliminate targets if it is out of drones.