Assignment 2: Question 4

April 2018

1 Termination

With the introduction of moving obstacles, oracles and charging stations the problem's termination condition changes. With infinite time, every oracle can be reached and spoken to. Thus the program's fail condition can be defined as:

```
    (identity is not known)
and (for all oracles, oracle is spoken to).
```

2 Reachability

In this redefinition of the problem the concept of **reachability** is introduced. In question 3, this concept ignored with by use of BFS from the starting location to find all oracles. As such only reachable oracles and charging stations are known to the program. However, now all oracles **can** be reached thus it is desirable to know the locations of all oracles. As such, the oracles are instead found by scanning the available grids. An oracle's state is asserted using the dynamic predicates:

```
oracle_pos/2
oracle_visited/1
unreachable_oracle/1
```

3 Dynamic A*

With moving blocks, the path chosen by A* may be unavailable by the time our Agent has reached the destination. To accommodate this, the program optimistically follows the path taken by A* and when the predicate:

```
take_path(Agent, Path) :-
    query_world( agent_do_moves, [Agent,Path] ).
```

fails a new path is calculated and attempted. From this arises some new states that must be addressed.

- If no path can be found by A* an oracle is flagged as unreachable.
- If an attempt to interact with an object fails the location of the object is retracted and rediscovered (This search is done by scanning tiles radiating out from the Agent's current location).
- If all un-visited oracles are deemed unreachable, they are unflagged and the algorithm tries again. While provides the correct behaviour with respect to the termination condition above, it is assumed the marker does not have "infinite" time to test the program. As such the unflagging can only happen a maximum of 10 times before the program will fail (This is primarily to deal with boards where most the tiles are obstacles).

Finally, an optimisation is applied to the A* search such that the depth of search is limited to 1.5 times the Manhattan distance to the goal. This prevents the program from wasting time searching the whole space when an unreachable object is close by.