F29AI
Artificial Intelligence and Intelligent Agents

PDDL Assignment

Authors

Tommy Ian Lamb - H00217505

Craig James Duffy - H00235298

Dillon Jabez Fearns - H00216974

November 13, 2017

1 Solution and Interesting Features

Our solution was developed using the latest version of the Fast Downward planner for PDDL. We make extensive use of PDDL features which may not be supported in all planners and, as such, cannot provide any guarantee that our solution will work efficiently, or reliably across planners.

1.1 Solution

Our modelling of the problem can be found in the *ship.pddl* file and begins with the domain and requirements, types - of which we have made extensive use, predicates, and derived predicates - a feature only supported by Fast Downward as best we could tell, and finally, all the actions.
Actions Include:
move_using_door: Allows Humanoids to move around a deck via doors.
move_using_lift: Allows Humanoids to move between decks via lifts.
move_ship: Moves the Ship between planets.
teleport_to_planet Teleports all light objects to the planet using the transporter.
teleport_from_planet: Teleports all light objects from the planet using the transporter.
fix_ship: Repairs the ship.
fix_transporter: Repairs a damaged transporter.
board_shuttle: Allows a humanoid to board a shuttlecraft.
alight_from_shuttle: Allows a humanoid to alight from a shuttlecraft.
send_shuttle_to_planet: Allows a humanoid pilot to fly a shuttlecraft to a nearby planet.
return_shuttle_to_ship: Allows a humanoid pilot to fly a shuttle back to the ship.
pickup_light_object: Allows a humanoid to pick up a light object.
drop_light_object: Allows a humanoid to drop a light object.
pickup_heavy_object: Allows a robot to pick up a heavy object.
drop_heavy_object: Allows a robot to drop a heavy object.
recharge: Allows a robot to be recharged.
heal: Allows a person to be healed.
raise_shields: Raises the ships shields.
drop_shields: Lowers the ships shields.

1.2 Interesting Features

There are a couple oddities relating to how things are implemented; this section aims to outline, and discuss these so that the code is easier to understand:

Shuttles are Both Objects and Places:
In order to model the fact that shuttlecraft can be both inside another place (such as inside a shuttle bay) and can have other objects, and humanoids inside of it. We have modelled the shuttlecraft as both objects and places, therefore, when we talk about the shuttlecraft as an object (such as checking if it is in a shuttle bay) then it will be treated as that while also ensuring we can treat it as a location when needed without the need for having two separate objects or extensive checks during these events.

We Never Check Plasma is not on a Shuttlecraft:
We deemed it unnecessary in our implementation to check if the Plasma object was present on a shuttlecraft before attempting to take off. This is due to the fact that our *board_shuttle* action only works on Humanoids, as such, there should never be plasma on the shuttlecraft. That is, of course, unless you dictate it in your initial state.

People and Robots are light objects:
Both people are robots are modelled as humanoids, and humanoids are modelled as light objects. This decision was made as it allows us to only check one condition (if the item is a light object) when using the transporters, we also envision the robots to be humanoid in shape.

Decks and Lifts:
We have decided to model the ship as a series of rooms connected to decks, and decks connected to lifts. This allows us to go from A to B by only checking if there is a path between these two areas. The planner does not need to output *"Move from A to D to E to B"* it can simply say *"Move from A to B"* provided the two areas are connected. In order to do this, we model a *lift_shaft* and model that if the room is connected to a deck, and if the deck has a lift shaft and is the destination deck has a lift shaft, and the destination room is connected to that deck then A and B are connected.