Franklin Camacho

Prolog Challenge 3

Task 4

New Cone Type

The first thing to do when beginning to model this new block world is add new facts for the added cone type (cone(c1)), along with editing the intial_state to include these new cones on the table itself. Since these cones will have colors too, we must edit the color part of the facts, something like “ color(c1, yellow) “ should be added in for each cone.

In the rules section of our code, we would have to add a validate rule so that a cone can be placed on top of a block, but a block or cone cannot be stacked on top of another cone. We would also need to edit the clear, holding, is_on rules as well, since we now have to add other possibilities for when you are holding a cone, or a cone is on top of a block. The same would have to be done with the perform command, since we have to add new possibilities for picking up cones, stacking cones, and unstacking them.

For the parsing section of our code, we would just have to add a new noun line for the cone type, something similar to this: noun(n(cone)) → [cone].

We would need to add a few extra lines to our reference resolution part of the code, so it includes the new cone type as well as the original block type. A way we might do this is adding a “;” part at the end of each statement, then copying the original block resolve_ref, just replacing “Block” with “Cone”.

Ordering Objects:

When considering the location of each block/cone in relation to the rest, we would have to add a few extra parts to our parsing, for “to”, “right” and “left”. For the natural language generation portion, we would need to add new statements, so it would be able to say sentences like “The small red block is to the right of the large yellow block.” and other similar statements. We would also need to add a few new facts, since we need to represent what block is to the left of another, and same for right. Using these facts we can make some rules to change the position of a block relative to the others when they are moved, so if a green block to the left of a red block is placed to right of the red block, the world should update so it shows what the new state of the world is. Another change would be in the reference resolution, since we would have to add the possibility of a new adjective in “left” or “right”.

Hypothetical Questions:

To be able to ask hypothetical questions about the block world, we would need to add a new command in the main loop, based on the initial state. With this, you could ask “can the blue cone be stacked on the small red block?”, and the system would check it based on the initial state of the world, and check the rules, seeing if this is a possibility. If the cone was able to be placed on the block, it would return true or yes, and if not, it would return false or no.

Refer to Objects Based on Location:

Using some of the rules from before, we would need to add some commands to the main loop section of the code, using some of the new resolve_ref lines for the possibility of a “left” or a “right”. We would also have to add new parts to our natural language generation part of the code, since now we would have to write “to the right of” or “to the left of”. Possibly looking like: nlg(left(Block, Target)) :- nlg(Block), write(“ is to the right of “), nlg(Target), nl. Something similar can be done if the block is to the right instead.