Fluents Used:

agent(X, Y, result(A,S),Ships, Capacity):-

This is our *successor-state-axiom* where we check all the possible actions that the agent can take. X and Y are the locations of the agent (coast-guard) on the grid. result(A, S) helps in constructing the final path of the goal. A is the action taken at each step. Capacity is the capacity of the coast guard.
For each new call generated from the fluent, we update one of the X, Y, and Ships variables. result(A, S) is updated according to the action taken.
For action A to be set down, the coast guard has to be at an X that is less than H (grid height). Similarly, for action down, X has to be larger than 0 (grid top). For moving left, Y has to be larger than 0, and for moving right, Y has to be smaller than W (grid width).
Also, to perform the action pickup, we first need to check that the coast guard is at one of the ships' locations and then we update the remaining list of ships. Finally, for the drop action, we check the coast guard is indeed carrying passengers and that he is currently standing at a cell containing a station.

update_ship(Member, Ships, Ships2):-

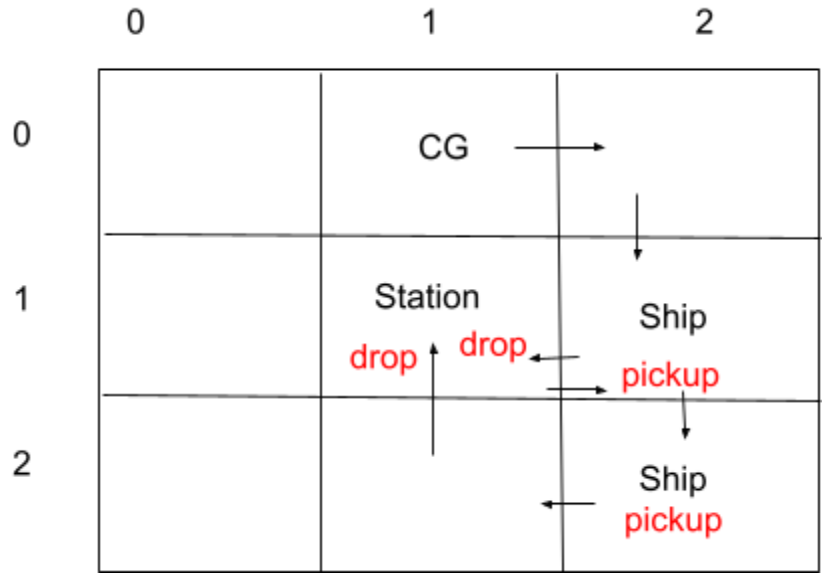
Member is the current location of the coast guard [X,Y] and Ships is the list of ships in the knowledge base. First, we check if Member is in Ships. If true, we remove Member from Ships and return the new list of ships in a new variable, Ships2.

Both **goal_helper2(S) and goal_helper1(X,L)** are helper methods for fluent 'goal(S)' to make use of call_with_depth_limit method to avoid out of stack error.

goal(S):- agent(X,Y,S,Ships,Capacity),station(X,Y), Ships=[],capacity(Capacity).
This fluent takes only one parameter which is a sequence of actions. The fluent checks for the final state that the agent should be in whenever he is in a goal state.
Condition #1:
The agent is in a goal state when he is at a station after he dropped the last remaining passenger. Therefore the final location of the agent should be the location of a station. agent(X,Y,S,Ships,Capacity) , station(X,Y). Anding these 2 fluents together checks that the final location of the agent (X,Y) is also the location of a station.
Condition#2:
Initially, the agent stores an array of all ships that he needs to rescue which is included in the knowledge base. Whenever the agent rescues the last passenger on this ship he removes this ship from the array of remaining ships. Therefore at a goal state, the array of remaining ships to rescue should be empty. This condition is checked by checking that Ships=[].
Condition #3:
Finally, at a goal state, the agent should drop all passengers on board and his capacity should return back to full capacity. Therefore to ensure that the agent dropped all passengers at the station (not just arrived at the station) we check that the final capacity of the agent is his full capacity. The knowledge base has the predicate capacity(FullCapacity), therefore the term *capacity(Capacity)* simply checks if the current capacity: *Capacity* is the *FullCapacity,*

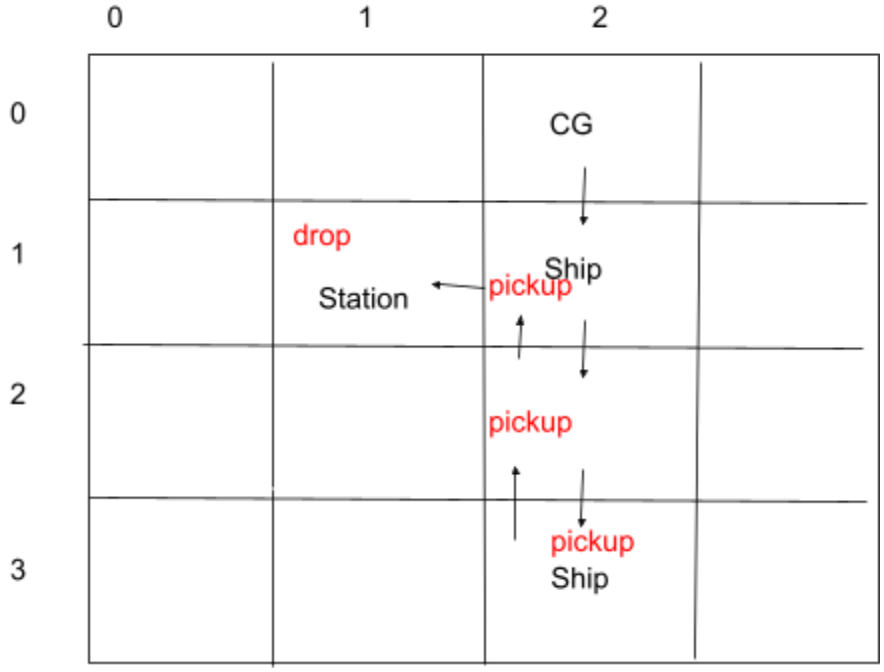
(with L set to 1)

KB output: (takes around 8 min 40 sec)



S = result(drop, result(up, result(left, result(pickup, result(down, result(right, result(drop, result(left, result(pickup, result(down, result(right, _G1892)))))))))))

KB2 output: (takes 15sec)



S = result(drop, result(left, result(pickup, result(up, result(up, result(pickup, result(down, result(down, result(down, _G3252373)))))))))

Note: we set **L in the call_with_depth_limit to 10** to make it faster, giving us same output but with shorter time. First output took around 8 minutes and second output took 15sec.

S = result(drop, result(up, result(left, result(pickup, result(down, result(right, result(drop, result(left, result(pickup, result(down, result(right, _G1892))))))))))) for KB.

S = result(drop, result(left, result(pickup, result(up, result(up, result(pickup, result(down, result(down, result(down, _G3253421))))))))) for KB2.

Both lead to a goal.

Note:
It might take up to around 20 minutes on another laptop. However, it took less than 10 minutes on two members' laptops of this team.