Automated Coffee Shop

The coffee shop has two robots, one barista and one waiter. The barista is in the bar and its role is to prepare cold and warm drinks with different preparation times. When the drinks are ready, the waiter robot can carry them either using its gripper or using a tray. Up to three drinks can be carried using the tray, whereas the waiter's moving speed is reduced in order not to pour out the drinks. The waiter moves at 2 meters per time unit; 1 meter per time unit if it is using the tray. Then, it needs to clean tables, which takes 2 time units per square meter to clean a table in order to clean the table it needs to leave the tray on the bar. In the coffee shop, there are four tables and a bar, shown in figure () and the third table is 2 square meters, others are 1 square meter. The distances are 1 meter apart from any other, and the bar is 2 meters away from the first two tables. There are four different planning problems to solve, but there are also four extensions which are implemented to the planning problem.

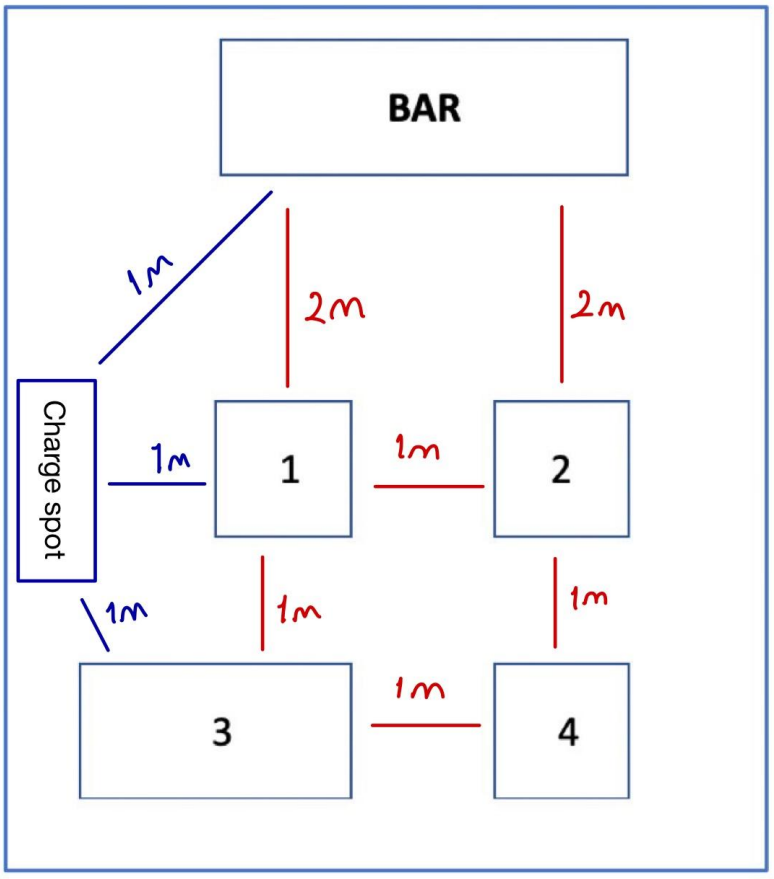

figure: coffee shop layout

1. Warm drinks cooling down: In this option, the waiter has to serve the warm drinks before they get cold. A warm drink takes 4-time units to become cold, otherwise, the drink can not be served to the customer.

2. Two waiters: Each waiter needs to be assigned different tables and can just serve that table.
3. Finishing the drink: The robot needs to clean the table when the customers finished their drinks and leave, which takes 4-time units.
4. Serving food: The waiter has to serve biscuits to the customers who ordered cold drinks. Biscuits should be served after the cold drink.

Waiter taking and giving drinks

For taking the drinks, the first preconditions are the drink is prepared (ready ?d - drink), preparation time (ready-time ?d - drink), and also the waiter should be at the bar using the predicate (at-rob ?r - waiter ?l - location) with holding the drink with the gripper (hold-drink ?g - gripper ?d - drink ?w - waiter) or without holding the tray (hold-tray ?g - gripper ?t - tray ?w - waiter), this is a precondition of action take-tray. If it decides to take drinks on the tray (on-tray ?t - tray ?d - drink), the number of drinks on the tray is counted until 3 using the function (tray-level ?t - tray).
Then, it can start moving, either slowly or fast depending on its decision of using the tray or gripper with the help of these preconditions (fast-moving ?w - waiter ?from ?to - location), (slow-moving ?w - waiter ?from ?to - location) inside the processes move-slow/fast. For giving the drinks, the waiter needs to be at the table, and also the customer should be still waiting at the table. So, it can serve the drink to the customer. In this case, if the tray is used in the serving process again, the drink that is served is counted in order to understand how many drinks are still on the tray. After finishing all the drinks on the tray, it leaves the tray on the bar with the help of action give-tray.

Two waiters

Two waiters are added as an extension, in order to do this a charge spot is added to the domain as a location type and waiter locations are initialized using (at-rob wairob1 chargespot), (at-rob wairob2 chargespot).
The distance between the charge spot to the bar side, first and third table is 1 meter. As it is shown in the figure (2.1). For initialization, the following equations are used.
(= (distance chargespot barside) 1)
(= (distance chargespot table1) 1)
(= (distance chargespot table3) 1). So, the waiters can start their day at the charge spot. Then, one of them can go to the bar to take the drinks, and at the same time, the other waiter can go for cleaning. This action decreases the total time of solving the problems. In order to prevent going to the same table for the orders, at least one table the waiter is assigned using the order predicate (order ?w - waiter ?t - table). For instance, in the first problem, the first waiter is assigned to the second table and so on.