INSTITUTO TECNOLÓGICO DE ESTUDIOS SUPERIORES DE MONTERREY

Ingeniería en Tecnologías Computacionales (ITC)

Modelación de sistemas multiagentes y gráficas computacionales (TC2008B.1)

Actividad integradora

David Eduardo Santiago Sánchez A01733753

29 de noviembre de 2021

Puebla, Pue.

Introducción

La actividad integradora de la presente Unidad de Formación solicitaba modelar el comportamiento de agentes tipo robots para ordenar las cajas dentro de un almacén.

Dentro de las características específicas de los agentes robots es posible destacar que pueden moverse en cuatro direcciones, necesitan estar al lado de una caja para poder levantarla, pueden saber si están cargando una caja, detectan qué objetos tienen a su alrededor y que pueden formar pilas de hasta 5 cajas.

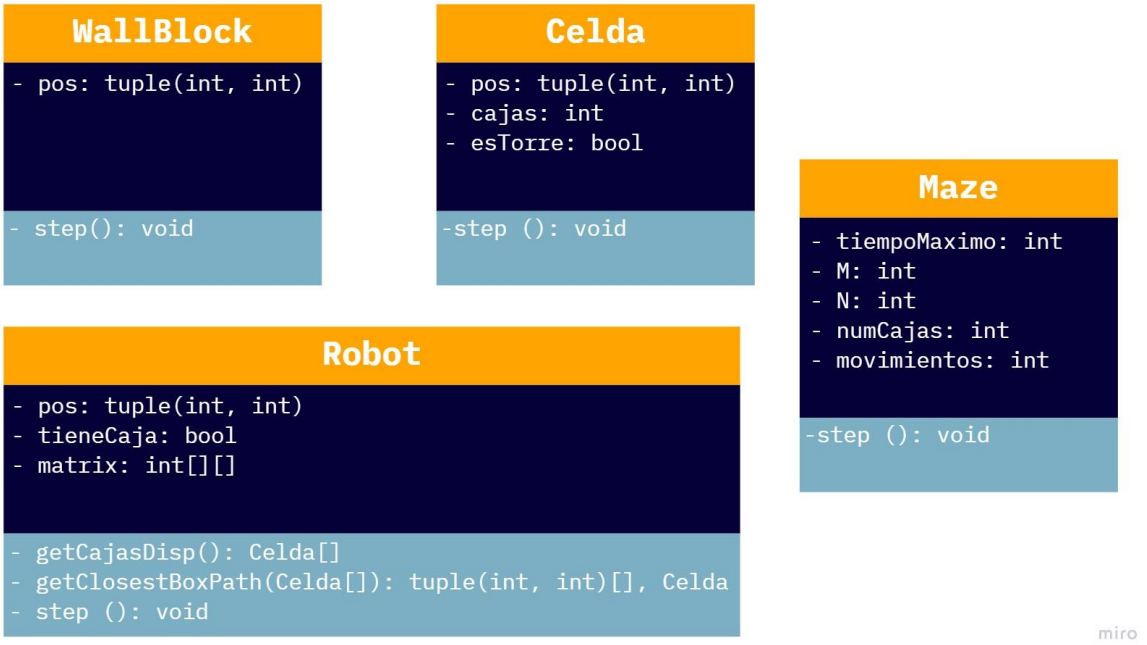
Se solicitó que la simulación realizada debía inicializar un total de *K* cajas, todas a nivel del piso. De igual manera, los agentes debían estar posicionados en lugares aleatorios en donde no existieran cajas u otros obstáculos.

Una vez iniciada la simulación, se debe respetar el tiempo límite indicado, recolectando mientras sea posible, el tiempo necesario para que se apilen todas las cajas en pilas de máximo 5 cajas y el número de movimientos que realizaron los robots.

Proceso de solución

Diagramas de clases

Primeramente, fue necesario definir las clases que representarían la situación solicitada. Por lo tanto, se definieron las siguientes clases:



Para representar las paredes y obstáculos dentro del almacén, se usan agentes WallBlock, cuyo único propósito es ser posicionados en el tablero para ser vistos por otros agentes.

Los agentes celda son los encargados de vigilar las posiciones de las cajas mediante su atributo cajas, así mismo, al inicio de la simulación se determinará qué celdas serán los puntos de destino para apilar las cajas del almacén, eligiendo posiciones donde ya esté posicionada una caja.

Los robots solicitados están modelados mediante la clase con el mismo nombre. Estos agentes tienen una matriz para que tengan la capacidad de reconocer si existe un obstáculo en una posición o si es transitable. Así mismo, tiene el atributo *tieneCaja*, tal como lo especifica el planteamiento del problema.

Finalmente el modelo se denomina Maze, esta clase tendrá información de la simulación, tal como el tamaño del almacén, el número de cajas, la matriz de posiciones transitables y se encargará de posicionar y llamar los métodos de los agentes dentro de la simulación.

Protocolo de agentes

Ahora bien, es posible observar que el único agente con métodos específicos es el agente Robot, por lo que para poder entender dichos métodos se realizaron los protocolos entre agentes mostrados en el *Anexo 1*.

En cada paso de la simulación el Robot llama a un método *getCajasDisp*, el cual devuelve una lista de agentes celda. En caso de que el agente no esté cargando una caja, se devolverán todas las celdas que tengan encima una sóla caja, mientras que si ya está cargando una caja, se devolverán las celdas que fueron marcadas como torres al inicio de la simulación y que cuenten con menos de 5 cajas apiladas.

Una vez obtenida la lista de celdas, el Robot ejecuta un método *getClosestBoxPath*, pasando como argumento la lista de celdas obtenidas. Este método itera sobre todas las celdas obtenidas y obtiene las posiciones adyacentes transitables, calculando el camino más corto a cada una de esas adyacencias mediante una búsqueda A*.

Después de calcular estos caminos, guarda la longitud del más corto, de manera que este camino será usado por el agente para desplazarse por el tablero hacia la posición más cercana según el propósito del agente: recoger o apilar. Cada vez que el agente se mueve, se sumará uno al contador de movimientos del modelo.

Finalmente, cuando el agente llega a su destino, se procederá de diferente manera según la acción. Si el agente desea recoger una caja, se cambiará su estado para indicar que ya tiene una caja, y la celda de la cual se ha recogido la caja, disminuirá su contador de cajas a 0. Por otra parte, si el agente desea apilar una caja, se le sumará uno al contador de cajas de la celda y el indicador del agente *tieneCaja* cambiará a falso.

El proceso de relaciones anterior seguirá corriendo hasta que se alcance el tiempo máximo o bien, se haya completado la tarea con éxito. Para que la tarea sea considerada como exitosa, no deberá encontrarse ninguna caja fuera de las celdas designadas como torres y tampoco deberá existir algún Robot cargando una caja.

Posibles mejoras

Si bien la solución reduce el tiempo y el número de pasos de los robots para completar la tarea solicitada al implementar una técnica de pathfinding en los agentes en lugar de implementar una técnica de movimientos aleatorios, es posible mejorar la implementación al elegir de otra manera las posiciones finales de las torres.

Si se implementara un algoritmo para realizar clusters dependiendo de la distribución de las cajas en el tablero, sería posible elegir de mejor manera las celdas que funcionarán como torres para cada escenario, reduciendo el desplazamiento necesario para llevar una caja a su destino.

Por otro lado, es posible mejorar el comportamiento de los robots, pues en la solución realizada, dos robots podrían dirigirse a recoger la misma caja, de manera que la tomará el robot que llegue primero. Para evitar el desplazamiento del segundo robot, debería implementarse una relación entre agentes de tipo Robot para indicarle a los demás quién se encargará de recoger qué caja.

Mediante estas dos mejoras, la simulación reduciría al máximo el número de pasos de agentes y el tiempo necesario para completar la tarea.

Anexo 1: Protocolo de agentes