Actividad Integradora: Robots

Carolina Ortega Barrios A01025254

Ian Seidman Sorsby A01028650

Ximena González Ibarra A01028604

Instituto Tecnológico de Estudios

Superiores de Monterrey, México, México

Fecha de entrega: Miércoles 24 de noviembre de 2021

Diagramas de Clases:

Clase Robot

Atributos	Métodos
Condition (String: HasBox, NoBox)	boxNeighbor(self) - checa vecinos y si uno es caja regresa su posición como tupla
Direction (Int: 0-4)	canGrab(self) - checa vecinos y si uno es caja que no tenga condición “Goal” regresa True
Moves (Int)	robotN(self) - checa vecinos y si uno es robot regresa su posición como tupla
Box (Box Agent)	noObst(self) - checa vecinos y si está completamente libre de agentes regresa True
Position (Tupla)	randomMove(self) - mueve al agente a otra parte del grid dependiendo en su atributo Direction No regresa algo
	randomMoveObst(self) - mueve al agente de forma aleatoria (solo a espacios vecinos sin agentes presente) No regresa algo
	destMove(self, xOY, dist) - mueve al agente hacia una coordenada destination dependiendo en los variables xOY y dist No regresa algo
	grab(self) - cambia la condición self.condition a “HasBox” y mueve el agente de Caja a la posición de self.pos, también le asigna a self.box el agente de Caja y le cambia la condición de esa caja a “Dynamic”
	dropBox(self) - cambia la condición self.condition a “NoBox”, cambia la condición de self.box.condition a “Goal” y mueve self.box a la posición de destination

Clase Caja

Condition	“Static”, “Dynamic” o “Goal”
Position	Tupla de coordenadas

Protocolo de Agentes:

1. Sí el robot tiene caja y se encuentra en alguna de las dos celdas contiguas a la celda destino (o sea (1,0) o (0,1)), el robot deposita la caja en la celda destino (0,0)
2. Sí el robot tiene caja y no hay algún obstáculo en su camino, se hace una resta de las coordenadas destino menos las coordenadas de las posiciones para que el robot se dirija de forma directa.
3. Sí el robot no tiene caja, hay una caja en alguna celda y esa caja no tiene como condición “Goal”, el robot debe tomar la caja y llevarla a su misma celda (posición)
4. Sí el robot no tiene caja y hay un obstáculo en su camino, el robot crea una lista de tuplas con las posiciones disponibles y de esa misma lista toma una posición random.
5. Sí tiene caja y además hay un obstáculo en el camino, ocurre lo mismo que el paso anterior, se crea una lista de tuplas con las posiciones disponibles y de ahí se selecciona una posición random.
6. Si no se cumple alguna de las instrucciones anteriores, el robot se mueve de manera aleatoria.

Estrategia Cooperativa para la Solución del Problema

Tenemos 3 propuestas de mejora para que el modelo funcione de manera mucho más eficiente, claro que esto incluye la comunicación que pueden tener los robots entre sí.

1. La primera es que se vayan guardando las posiciones de los obstáculos (en este caso las cajas que hay en el grid) para que así los robots sean capaces de esquivar las posiciones en donde haya una caja obstáculo (en el caso de que cumplan la condición de que tienen una caja), y que los robots que estén

disponibles sepan exactamente a donde ir en lugar de escoger una posición random hasta encontrar una caja.

2. Otra opción que pensamos es que los robots vayan guardando las posiciones visitadas, para que el robot le pueda dar prioridad a las otras celdas.
3. Lo ideal sería que los robots puedan darle prioridad a los robots que tienen una caja y van al destino, para que así puedan pasar mucho más rápido sin enfrentarse a más obstáculos.