Proyecto de Simulación
Agentes

Carlos Aryam Martínez Molina

Curso: 2019-2020

Índice

1. Datos generales del estudiante

- Nombre: Carlos Aryam Martínez Molina
- Grupo: C-411
- Provincia: Matanzas

2. Orden del Problema Asignado.

Orden brindada por el colectivo de Simulación

3. Principales Ideas seguidas para la solución del problema

Cada elemento del ambiente es colocado de forma aleatoria en una matriz, teniendo en cuenta sus restricciones como por ejemplo que los corrales deben estar en casillas adyacentes. La variacion del ambiente es realizado guardando en una structura todos los elementos actuales en la matriz limpiando la matriz y luego colocando estos mismos elementos en posiciones random chequeando que se cumplan con las restricciones primero los corrales adyacentes y si había un niño en el corral este se pone en el mismo corral, los demás elementos son ubicados de forma aleatoria incluyendo el robot. Como idea para el robot este tiene dos políticas la primera camina hacia el niño más cercano y si se encuentra con basura en sus adyacentes la limpia y luego sigue, una vez con el niño cargado lo lleva hacia el corral más cercano con la misma idea, si encuentra una basura en sus adyacentes la limpia y después sigue, esto explota la idea de no dejar que el ambiente se llene de basura y recoger siempre que tenga la oportunidad sin desviarse de su objetivo principal. La segunda política busca el niño más cercano sin importar lo que encuentre en el camino y llevarlo al corral sin importar lo que encuentre en el camino, y despues que todos los niños estén en sus corrales recoge toda la basura, esta forma explota la idea de mientras más tiempo esten los niños fuera del corral más basura generan por eso se recogen estos lo antes posible. Nota Cuando se dice sin importar lo que ecuentre en el camino se hace referencia a la basura, si es un obstáculo lo rodea o sea respeta las reglas del problema.

4. Modelos de Agentes considerados

Se consideraron dos tipos de agentes:

Reactivo:
Este agente recive información del ambiente y actua en consecuencia a esta, para lograr sus objetivos cambia de desición a medida que cambian las condiciones en que se encuentra. Por ejemplo la política uno se mueve en busqueda de un niño pero si encuentra basura en alguna de sus casillas adyacentes cambia la desición, recoge la basura y luego sigue buscando al niño, una vez que tiene al niño lo lleva hacia el corral con la misma idea si en el camino se encuentra con basura la limpia y después continua.

Proactivo:
Este agente es capaz de mostrar un comportamiento GOAL-DIRECTED, persiguiendo su objetivo trazado sin importar el cambio del ambiente. Por ejemplo la política dos se traza como meta encontrar al niño más cercano e ignora la basura que aprezca aunque le pase por arriba, luego de cargar al niño lo lleva al corrar ignorando la basura también y una vez todos los niños están en sus corrales entonces limpia la basura de la casa, esto evidencia el caracter proactivo el cual se traza la meta de buscar un niño y no cambia de objetivo no importa que cambios haya en el ambiente.

5. Ideas seguidas para la implementación

Para la implementación cree una clase por cada elemento del ambiente, incluyendo una clase para el propio ambiente, la clase del ambiente, consta de métodos como la generación random del tablero inicial que se llama *initalEnv* este se encarga de dar un tablero con los valores de entrada especificados de forma aleatoria en la matriz, tiene otro método *EnvironmentTurn* que ejecuta el movimiento de los niños y crea la basura aleatoria, tiene también un método *redistribucionRandom* que es el que se encarga de dado un estado del tablero devolver los mismos elementos del tablero pero en posiciones diferentes, la posición nueva que van a ocupar cada elemento en el tablero es elegida de forma aleatoria respetando la adyacencia de los corrales y demás reglas del ambiente. Por último consta de un método que se llama *checkDirtiness* que devuelv el porciento de suciedad en que está el tablero en ese momento, esto se verifica porque si tiene más del 60 la simulación se detiene y el robot es despedido.
Las clases Obstacle, Yard, Dirtiness y Empty solo tienen el símbolo con el que se las va representar en la matriz y el símbolo si están en conjunto con otra clase por ejemplo un Yard con un Children significa que el niño está en esa instancia del corral, pero no tienen métodos adicionales.
La clase niño consta de varios métodos, esta misma es la que se encarga de decidir si el niño se mueve o no, en la función *moving*, que está enlazada con *movObstacle* que si el niño se mueve hacia un obstáculo esta función se encarga de empujar todos los obstáculos en la dirección que se movió el niño si es posible y deja al niño en el lugar si no es posible empujar los obstáculos. Luego si el movimieto tuvo éxito se encarga de generar la basura correspondiente con el método. *toDirty*.
Luego en la clase robot implementé dos politicas de decisión del robot, la primera llamada $movPolicy1$ es la que se encarga de chequear todas las casillas adyacentes a las que se puede mover el robot y si este tiene un niño en una de las casillas adyacentes procede a cargar el niño, si hay basura se mueve a la casilla de la basura para limpiarla en el próximo turno si no, se mueve a la casilla en blanco o con corral vacío más cercana al niño más cercano del robot, estas opciones las realiza en ese orden de prioridad mencionado, todo esto se realiza en el método $movWithoutChildren$, una vez el robot tiene un niño cargado procede a llevarlo al corral más cercano limpiando todas las basuras que encuentre en las casillas adyacentes a el en su camino al corral proceso que se realiza en la función $movWithChildren$, cuando deposita el niño en el corral vuelve al estado anterior que usa el método $movWithoutChildren$ hasta que estén todos los niños en el corral en cuyo caso procede a limpiar la basura restante por orden de cercanía. La política 2 implementada en $movPolicy2$ es similar a la anterior pero cuando se encuentra el robot sin niño cargado busca el niño más cercano ignorando la basura, esto se hace en $movWithoutChildren2$ y lo pone en el corral más cercano ignorando la basura también con el método $movWithChildren2$ y una vez todos los niños están situados en el corral procede a limpiar toda la basura. Por último en el archivo *run.py* se crean 10 escenarios iniciales distintos y se corren 30 simulaciones por cada escenario una vez con una política y otra vez con la otra. En el método *simulation* se maneja la lógica del problema o sea primero ejecutar el turno del robot luego el del ambiente y cada cierta cantidad de turnos realizar el reordenamiento

aleatorio del problema. En cada turno se chequea que el robot no haya pasado de los 100 turnos, que no haya más del 60 por ciento de suciedad o que ya haya limpiado todo el ambiente para detener la simulación.

6. Consideraciones obtenidas a partir de la ejecución de las simulaciones del problema

Realizando las 600 simulaciones del proyecto podemos percatarnos que la primera política(la de limpiar la basura que se encuentra en el camino) es mejor que la segunda política(la de ignorar la basura hasta que todos los niños estén en un corral) en los casos en que el tablero está con más abundancia de basura porque en este caso los niños van a generar la cantidad de basura necesaria para parar la simulación antes de que al robot le de tiempo poner a todos los niños en el corral, y la segunda política es mejor en los casos en que hay menos abundancia de basura ya que si se pone a limpiar la basura sin recoger a los niños estos van a generar basura más rapido de lo que el robot la limpia por eso sería más eficiente hacer que los niños dejen de ensuciar y después limpiar y como no hay tanta abundancia de basura le da una apertura al robot, además que se aprobecha de los pasos dobles que da cuando carga un niño lo cual pierde si recoge basura en el camino. Esta observación anterior es basada en los datos recogidos en las simulaciones realizadas que me permiten llegar a esta conclusión que no es 100 por ciento verídica, aqui hablo de la mayoría de los casos de modo general, esto evidencia la importancia de la simulación de estos agentes que nos permiten llegar a conclusiones prácticas de forma eficiente, concreta y realista.

Resultados de las 600 simulaciones:

Ambiente 6x6,15 por ciento de basura ,20 por ciento de obstaculos, 1 niños y 6t
Reactivo
Porciento de casillas sucias medio: 0.0
Veces que el robot fue despedido : 0
Veces que el robot ubicó a todos los niños en el corral : 30

Proactivo
Porciento de casillas sucias medio: 0.0
Veces que el robot fue despedido : 0
Veces que el robot ubicó a todos los niños en el corral : 30

Ambiente 7x8,10 por ciento de basura ,10 por ciento de obstáculos, 3 niños y 5t
Reactivo
Porciento de casillas sucias medio: 0.0001728800790650504
Veces que el robot fue despedido : 3
Veces que el robot ubicó a todos los niños en el corral : 27

Proactivo
Porciento de casillas sucias medio: 6.386211940220424e-07
Veces que el robot fue despedido : 1
Veces que el robot ubicó a todos los niños en el corral : 29

Ambiente 10x10,15 por ciento de basura ,20 por ciento de obstáculos, 2 niños y 5t

Reactivo
Porciento de casillas sucias medio: 1.5060685770586133
Veces que el robot fue despedido : 14
Veces que el robot ubicó a todos los niños en el corral : 16

Proactivo
Porciento de casillas sucias medio: 7.67875462770462e-06
Veces que el robot fue despedido : 5
Veces que el robot ubicó a todos los niños en el corral : 25

Ambiente 7x8,10 por ciento de basura ,10 por ciento de obstáculos, 5 niños y 10t
Reactivo
Porciento de casillas sucias medio: 3.0450044748639424
Veces que el robot fue despedido : 19
Veces que el robot ubicó a todos los niños en el corral : 11

Proactivo
Porciento de casillas sucias medio: 0.1450771914928087
Veces que el robot fue despedido : 15
Veces que el robot ubicó a todos los niños en el corral : 15

Ambiente 12x8,40 por ciento de basura ,10 por ciento de obstáculos, 5 niños y 10t
Reactivo
Porciento de casillas sucias medio: 37.36671816538243
Veces que el robot fue despedido : 30
Veces que el robot ubicó a todos los niños en el corral : 0

Proactivo
Porciento de casillas sucias medio: 41.14572745941889
Veces que el robot fue despedido : 30
Veces que el robot ubicó a todos los niños en el corral : 0

Ambiente 15x15,5 por ciento de basura ,5 por ciento de obstáculos, 2 niños y 10t
Reactivo
Porciento de casillas sucias medio: 3.0763871500061617
Veces que el robot fue despedido : 29
Veces que el robot ubicó a todos los niños en el corral : 1

Proactivo
Porciento de casillas sucias medio: 0.8125839100943671
Veces que el robot fue despedido : 26
Veces que el robot ubicó a todos los niños en el corral : 4

Ambiente 20x15,20 por ciento de basura ,10 por ciento de obstáculos, 10 niños y 15t

Reactivo
Porciento de casillas sucias medio: 60.18407323397696
Veces que el robot fue despedido : 30
Veces que el robot ubicó a todos los niños en el corral : 0

Proactivo
Porciento de casillas sucias medio: 57.02280078704159
Veces que el robot fue despedido : 30
Veces que el robot ubicó a todos los niños en el corral : 0

Ambiente 5x5,50 por ciento de basura ,50 por ciento de obstáculos, 3 niños y 5t
Reactivo
Porciento de casillas sucias medio: 0.2362065576016903
Veces que el robot fue despedido : 4
Veces que el robot ubicó a todos los niños en el corral : 26

Proactivo
Porciento de casillas sucias medio: 2.2528070211410522
Veces que el robot fue despedido : 11
Veces que el robot ubicó a todos los niños en el corral : 19

Ambiente 10x5,10 por ciento de basura ,5 por ciento de obstáculos, 1 niños y 50t
Reactivo
Porciento de casillas sucias medio: 0.0
Veces que el robot fue despedido : 0
Veces que el robot ubicó a todos los niños en el corral : 30

Proactivo
Porciento de casillas sucias medio: 0.0
Veces que el robot fue despedido : 0
Veces que el robot ubicó a todos los niños en el corral : 30

Ambiente 15x15,25 por ciento de basura ,20 por ciento de obstáculos, 15 niños y 12t
Reactivo
Porciento de casillas sucias medio: 59.212726421240305
Veces que el robot fue despedido : 30
Veces que el robot ubicó a todos los niños en el corral : 0

Proactivo
Porciento de casillas sucias medio: 60.000109300431276
Veces que el robot fue despedido : 30
Veces que el robot ubicó a todos los niños en el corral : 0

Promedio de todas las simulaciones
Porciento de casillas sucias medio: 59.82854504546346
Veces que el robot fue despedido : 307
Veces que el robot ubicó a todos los niños en el corral : 293

Nota: Para correr el código solo es necesario hacer python run.py