Informe del Proyecto Agentes. Simulación. Curso 2020-2021

Leonel Alejandro García López L.GARCIA3@ESTUDIANTES.MATCOM.UH.CU
Grupo C412

Tutor(es):
Dr. Yudivián Almeida Cruz, *Facultad de Matemática y Computación, Universidad de La Habana*
Lic. Gabriela Rodriguez Santa Cruz Pacheco, *Facultad de Matemática y Computación, Universidad de La Habana*
Lic. Daniel Alejandro Valdés Pérez, *Facultad de Matemática y Computación, Universidad de La Habana*

Tema: Simulación, Eventos Discretos

1. Introducción

Como proyecto se propuso simular un ambiente donde intervienen agentes correspondiente a una casa. El ambiente es discreto y tiene la forma de un rectángulo de N, M . El ambiente es de información completa, por tanto todos los agentes conocen toda la información sobre el agente. El ambiente puede variar aleatoriamente cada t unidades de tiempo. El valor de t es conocido. Las acciones que realizan los agentes ocurren por turnos. En un turno, los agentes realizan sus acciones, una sola por cada agente, y modifican el medio sin que este varíe a no ser que cambie por una acción de los agentes. En el siguiente, el ambiente puede variar. Si es el momento de cambio del ambiente, ocurre primero el cambio natural del ambiente y luego la variación aleatoria.

En una unidad de tiempo ocurren el turno del agente y el turno de cambio del ambiente. Los elementos que pueden existir en el ambiente son obstáculos, suciedad, niños, el corral y los agentes que son llamados Robots de Casa.

1.1 Modelado e Implementación

El modelo empleado en la simulación, se baso en 3 módulos principalmente *commons*, *environment* y *agents*. En el modulo de ambiente se encuentra una implementación de *House* que no es mas que el ambiente en donde se desarrolla la simulación, contiene todas las funcionalidades necesarias para construir un campo aleatorio y manejar los distintos cambios que los objetos realizan, así como hacer efectivas las acciones de los agentes en el terreno.

Para construir un terreno es posible especificar la cantidad de obstáculos (o en por ciento), la cantidad de suciedad inicial (o en por ciento), la cantidad de bebes, un modelo para crear a los objetos de tipo bebe, un modelo de agente (con el cual se generan agentes), el tamaño entrado como tupla ($Dimension(cols, rows)$), el tiempo t en el que demora en variar el ambiente.

La clase *House* también se encarga de gestionar la variación que da lugar en el tablero cada t momentos, donde asumí como variación cambiar a los chicos que están fuera de los corrales, recolocar los corrales en nuevas posiciones(adyacentes entre si, de igual forma), reposicionar las suciedades, cambiar el lugar del o los agentes asi como los obstaculos, todo esto aleatoriamente.

En el método *turn_cycle* se encuentra el ciclo principal de nuestra simulación donde se le especifica si se dese paso a poso y si quiere una salida verbosa, en este ciclo como dice su nombre se suceden los turnos y el objetivo del Robot de Casa es mantener la casa (a.k.a el ambiente) limpia. Se considera la casa limpia si no más del *60 %* de las casillas vacías están sucias. Se sabe que si la casa llega al *60 %* de casillas sucias el Robot es despedido e inmediatamente cesa la simulación. Si el Robot ubica a todos los niños en el corral y el *100 %* de las casillas están limpias también cesa la simulación. Estos son llamados estados finales. Debe programar el comportamiento del robot por cada turno así como las posibles variaciones del ambiente. En este ciclo compruebo cada una de las condiciones anteriores antes de continuar al siguiente turno.

Luego está las implementaciones en el modelo *commons*, que refieren a lo común usado entre *environment* y *agents*. En este módulo encontramos las implementaciones útiles para:

- *Directions*: referido a la lista de direcciones nombradas, se puede obtener con *rdirs* una lista desordenada de direcciones, y otras mas funcionalidades necesarias en la dirección.
- *Coordinates*(**NamedTuple**): referido a la posición en el piso de la casa, con un método *on_direction* para obtener la correspondiente posición en la dirección especificada.
- *CellContent*(**Enum**): referido a los contenido de celdas, *Dirty*, *Empty*, *Obstácle* y *Playpen*.
- *Dimensions*(**NamedTuple**): referido a la dimensión del tablero.
- *distance*(**función**): función que describe la distancia, de forma que a partir del parámetro nombrado

extended se escoge uno u otro sistema de medición de distancias.

- *SimulationResult*(**NamedTuple**): tipo para almacenar de una manera estructurada la información obtenida de una simulación concluida.
- *AgentAction*(**Enum**) y *ChildAction*(**Enum**): para enumerar las posibles acciones que se pueden realizar y que en caso de escoger un acción el ambiente ejecuta dicha acción sobre el agente/chico en acción

Por último tenemos el módulo de agentes o *agents* que contiene las implementaciones para estos dada diferentes estrategías, percepción, etc. Todo agente hereda del tipo *HouseAgent* que contiene implementaciones útiles para sus descendientes pero no posee una implementación precisa del método empleado por el ambiente para cuestionar la acción a realizar en el turno actual, el método *Action*. Este básicamente contiene un campo *carrying* y un campo *coord* con la posición en la casa asociada.

Tipos Concretos de agentes:

DummyRobot: se mueve aleatoriamente (con doble paso si es posible) esto con ninguna influencia del ambiente, recoge al chico si está en la celda actual y deja al chico si se encuentra en la casilla de corral, así como también recoge basura si esta en el camino errático.

FocusedRobot: esta ya es una implementación más avanzada de robot de limpieza, este tiene una especie de objetivo o goal y hasta que no finalice llegando a su objetivo, no realiza otra tarea que no esté relacionada. (Avanza solo con un foco, si no tiene foco se queda en el lugar)

2. Simulación

2.1 Resultados Obtenidos