Proyecto de Simulación - Agentes

Eliane Puerta Cabrera - C412

Problema

Se tiene un ambiente, representado por un tablero rectangular de tamaño N×M, con el que interactúa un agente. 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. 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. El Robot de Casa parte de un posición aleatoria y es el el que realiza el primer turno. Igual, se especifica el valor del tiempo de unidades de cambio (t).

El objetivo del Robot de Casa es mantener la casa (a.k.a el ambiente) limpia. Se considera la casa limpia si el 60% de las casillas vacías no 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. En caso de que no se logre uno de los estados finales del ambiente, la simulación debe detenerse cuando hayan transcurrido 100 veces t. Debe programar el comportamiento del robot por cada turno así como las posibles variaciones del ambiente.

Ideas para solución del problema

El ambiente está formado por casillas donde puede haber uno de los siguientes elementos: corral, obstáculo, niño, suciedad, o estar vacía. Es un ambiente accesible y dinámico.

En cada momento, el ambiente puede estar en uno de tres estados:
− S_1: la suciedad es menor del 60% o hay niños fuera del corral.
− F_1: la suciedad sobrepasó el 60%, este es un estado final.
− F_2: la suciedad es 0 y todos los niños están en el corral, este también es un estado final.

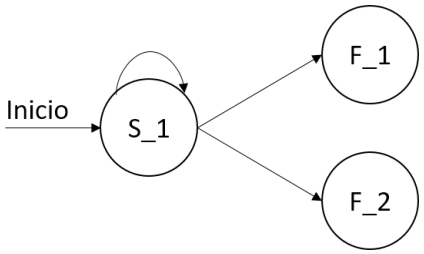


Figure 1: Esquema de estados del ambiente.

El agente es de tipo reactivo; en todo momento tiene conocimiento del estado del ambiente, así como de la posición donde está y los objetos a su alrededor. Se rige por la siguiente estrategia: si está parado en una casilla sucia la limpia, si se encuentra un niño inmediatamente lo recoge y lo lleva hacia el corral. Si un agente está cargando un niño, solamente lo suelta si la posición donde está parado es una casilla del corral vacía. Si no está ocupado limpiando o llevando un niño, entonces se mueve por el ambiente hasta que encuentre suciedad o un niño suelto.

Modelos de agente

Debido a que el agente interactúa con un ambiente dinámico, se implementó un modelo de agente puramente reactivo, siguiendo una arquitectura de categorización, y se implementaron tres modelos de agente con arquitectura por capas verticales.

Arquitectura de categorización:

Este agente realiza las acciones de acuerdo a las siguientes reglas:

1. si encuentra obstáculo → escoger nueva dirección

2. si tiene un niño cargado y llegó a una casilla corral → dejar el niño

3. si tiene un niño cargado → moverse

4. si no tiene niño cargado y encuentra basura → limpiar

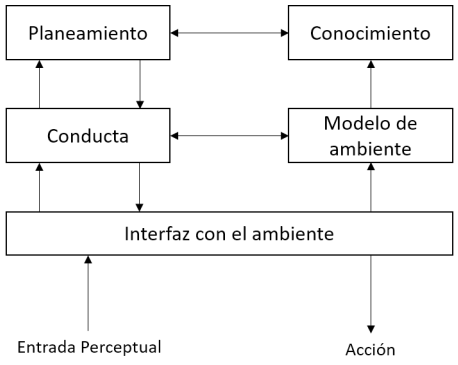
5. si no tiene niño cargado y encuentra niño → coger niño

6. moverse

en orden creciente de prioridad: $1 < 2 < 3 < 4 < 5 < 6$. Su movimiento es aleatorio.

Arquitectura por capas:

Consta de dos capas de control, una para el comportamiento reactivo y otra para la planificación. La diferencia entre estos agentes radica en la forma en que realizan la planificación.



Agente por capas 1:

En Modelo de ambiente guarda el ambiente en el que está, la posición que está ocupando en él y si carga un niño o no.

En Conocimiento guarda la última acción planificada.

La Conducta se determina siguiendo las siguientes reglas en orden ascendente:

1. si encuentra un obstáculo → cambiar dirección

2. si carga niño → hacer lo que indica la capa planeamiento

3. si no carga niño y hay basura en la posición → limpiar

4. si no carga niño y hay niño suelto en la posición → coger niño

5. si no carga niño y en la posición no hay nada → hacer lo que indica la capa planeamiento

El Planeamiento consta de las siguientes reglas:

1. si carga niño → buscar corral y moverse hacia allí

2. si no carga niño → buscar basura o niño más cercano y moverse hacia allí

Agente por capas 2:

El Modelo de ambiente y la capa Conducta de este son iguales al anterior. Este, en Conocimiento, guarda además la cantidad de niños que quedan sueltos.

Para el planeamiento se siguen las reglas:

1. si carga niño → buscar corral y moverse hacia allí

2. si no carga niño y niños sueltos > 0 → buscar niño más cercano y moverse hacia allí

3. si no carga niño y niños sueltos $= 0$ → buscar basura más cercana y moverse hacia allí

Agente por capas 3:

El Modelo de ambiente y la capa Conducta de este son iguales a los anteriores. Este, en Conocimiento, guarda, en lugar de los niños sueltos, la cantidad de suciedad que hay en el ambiente.

Para el planeamiento se siguen las reglas:

1. si carga niño → buscar corral y moverse hacia allí

2. si no carga niño y suciedad > 0 → buscar basura más cercana y moverse hacia allí

3. si no carga niño y suciedad $= 0$ → buscar niño más cercano y moverse hacia allí

La búsqueda y el movimiento hacia el objeto más cercano en estos agentes se realiza con una estrategia BFS.

Implementación

Ambiente: La implementación se realizó en C#. El ambiente está modelado con una clase `AmbientBoard` que guarda una matriz de `AmbientCell`. Cada `AmbientCell` guarda una referencia al objeto que está dentro de ella, dígase niño, corral, basura, obstáculo, agente, o casilla libre.

Objetos: Los objetos del ambiente están modelados con una interfaz `IAmbientElement`, que es implementada por las clases `Child`, `Filth`, `Obstacle`, `Playpen` y `FreeBox`, que representan los elementos del ambiente.

Agentes: Los agentes implementan las interfaces `IAmbientElement`, `IMoves` e `IAgent`, y heredan las funcionalidades comunes de una clase base `Agent`. Tienen una propiedad en la que guardan la posición del ambiente en la que están, y guardan también una referencia al niño que están cargando. Para el movimiento se consideraron solamente las direcciones arriba, abajo, derecha e izquierda, pero esto se puede cambiar fácilmente. El movimiento del primer agente es aleatorio hacia una de sus casillas adyacentes disponibles. El movimiento de los demás agentes se realiza con BFS para encontrar el camino más corto hacia la casilla buscada.

Simulación: Se tiene una clase `Simulation` que recibe una referencia al ambiente donde se quiere simular, y otra al agente que interactúa con dicho ambiente. Esta clase es la que maneja la transición entre el estado intermedio del ambiente y los estados finales. Maneja también la sucesión de los turnos de acción del agente y cambio natural del ambiente, y los turnos de variacion aleatoria del ambiente.

Programa: El programa completo tiene una lista con los agentes que se quiere estudiar y una lista con 10 ambientes generados aleatoriamente (con los parámetros de entrada especificados). Por cada agente se crea una simulación para cada ambiente, con un valor aleatorio de t, y se repite esta simulación 30 veces.

Resultados:

Luego de las 30 simulaciones de los agentes en cada ambiente se obtuvieron los siguientes resultados:

Agentes	Random	BFS	Niños Primero	Basura Primero
Suciedad(%)	29.20	20.25	30.90	19.58
Veces Despedido	76	49	66	50
Terminó Ok	60	73	62	72

A partir de los resultados obtenidos, se pudo apreciar que el primer agente, con arquitectura de categorización, tuvo, en promedio, un desempeño inferior al de los otros agentes. Fue despedido mayor cantidad de veces que los demás, lo que se traduce en que permitió que la suciedad sobrepasara el 60% en más ocasiones. Además, terminó correctamente, o sea, limpió todo y recogió todos los niños, en menos ocasiones también. Tiene también la tasa $\frac{ok}{despedido}$ más baja.

En los agentes con arquitectura en capas se observa un comportamiento mejor. No obstante, el agente que recoge primero los niños tuvo un desempeño mucho más pobre que los otros; su tasa $\frac{ok}{despedido}$ es la más baja de los tres y fue despedido un mayor número de veces.

El agente que mejor desempeño tuvo fue el que hace primero la tarea más cercana, lo que indica que en un ambiente tan dinámico como este, una estrategia de planeamiento tan rígida como las de los otros, no es la mejor forma de completar los objetivos del agente. Este fue el que mayor tasa $\frac{ok}{despedido}$ alcanzó. Fue despedido un menor número de veces y terminó

bien en más ocasiones. No obstante, el desempeño del agente que limpia la basura primero fue casi idéntico al de este, aunque mantuvo un menor porciento de basura en el ambiente.

Teniendo en cuenta el análisis de los resultados, el modelo de agente más adecuado para este ambiente parece ser el de arquitectura en capas, con la estrategia de buscar el objeto más cercano. No obstante, la rutina de generación de basura implementada puede no haber sido la que mejor modela este proceso en un ambiente como este.