Proyecto de Simulación y Programación Declarativa: Agentes

Richard García De la Osa C412

Git link: https://github.com/Regnod/Proyecto-Agentes.git.

Use cabal para el desarrollo del proyecto.

Orden del Problema Asignado

1. Marco General
   El ambiente en el cual intervienen los agentes 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. A continuación se precisan las características de los elementos del ambiente:

-Obstáculos: estos ocupan una única casilla en el ambiente. Ellos pueden ser movidos, empujándolos, por los niños, una única casilla. El Robot de Casa sin embargo no puede moverlo. No pueden ser movidos ninguna de las casillas ocupadas por cualquier otro elemento del ambiente.

-Suciedad: la suciedad es por cada casilla del ambiente. Solo puede aparecer en casillas que previamente estuvieron vacías. Esta, o aparece en el estado inicial o es creada por los niños.

-Corral: el corral ocupa casillas adyacentes en número igual al del total de niños presentes en el ambiente. El corral no puede moverse. En una casilla del corral solo puede coexistir un niño. En una casilla del corral, que esté vacía, puede entrar un robot. En una misma casilla del corral pueden coexistir un niño y un robot solo si el robot lo carga, o si acaba de dejar al niño.

-Niño: los niños ocupan solo una casilla. Ellos en el turno del ambiente se mueven, si es posible (si la casilla no está ocupada: no tiene suciedad, no está el corral, no hay un Robot de Casa), y aleatóriamente (puede que no ocurra movimiento), a una de las casilla adyacentes. Si esa casilla está ocupada por un obstáculo este es empujado por el niño, si en la dirección hay más de un obstáculo, entonces se desplazan todos. Si el obstáculo está en una posición donde no puede ser empujado y el niño lo intenta, entonces el obstáculo no se mueve y el niño ocupa la misma posición. Los niños son los responsables de que aparezca suciedad. Si en una cuadrícula de 3x3 hay un solo niño, entonces, luego de que él se mueva aleatoriamente, una de las casillas de la cuadrícula anterior que esté vacía puede haber sido ensuciada. Si hay dos niños se pueden ensuciar hasta 3. Si hay tres niños o más pueden resultar sucias hasta 6. Los niños cuando están en una casilla del corral, ni se mueven ni ensucian. Si un niño es capturado por un Robot de Casa tampoco se mueve ni ensucia.

-Robot de Casa: El Robot de Casa se encarga de limpiar y de controlar a los niños. El Robot se mueve a una de las casillas adyacentes, las que decida. Solo se mueve una casilla sino carga un niño. Si carga un niño puede moverse hasta dos casillas consecutivas. También puede realizar las acciones de limpiar y cargar niños. Si se mueve a una casilla con suciedad, en el próximo turno puede decidir limpiar o moverse. Si se mueve a una casilla donde está un niño, inmediatamente lo carga. En ese momento, coexisten en la casilla Robot y niño. Si se mueve a una casilla del corral que está vacía, y carga un niño, puede decidir si lo deja esta casilla o se sigue moviendo. El Robot puede dejar al niño que carga en cualquier casilla. En ese momento cesa el movimiento del Robot en el turno, y coexisten hasta el próximo turno, en la misma casilla, Robot y niño.

2. Objetivos

   El objetivo del Robot de Casa es mantener la casa limpia. Se considera la casa limpia si el 60% de las casillas vacías no están sucias.

Principales Ideas seguidas para la solución del problema.

La idea principal era mantener toda la información necesaria almacenada en un tablero y en listas auxiliares con los valores de los roboces, niños y objetivos. De esta forma al mover tanto los roboces como los niños podría considerar el ambiente alrededor de los ellos, al igual que al mover un obstáculo podría considerar la dirección y revisar si existe la posibilidad de movimiento. También tener en cuenta los estados de los roboces y de los niños para modificar el ambiente en correspondencia a los mismos y mantener un orden armónico del tablero.

El funcionamiento de la simulación corresponde a dos posibles ciclos, el primer ciclo es el que se encarga de los t-1 turnos de movimiento de los roboces y de los niños antes de la variación aleatoria del ambiente, en este se ejecutan las acciones de los agentes y de los niños luego. Después se toma el tablero modificado y se ejecuta otro turno de movimiento esta vez solo de una unidad de tiempo e inmediatamente después se genera la suciedad a partir de los movimientos de los niños. El segundo ciclo es correspondiente a esto último.

- primer ciclo: corresponde al movimiento de un turno de los roboces y los niños t-1 veces
- segundo ciclo: está formado por el primero más otro turno y la generación de suciedad.

Cada paso del primer y segundo ciclo es impreso en la consola para tener una visualización de lo que esta ocurriendo.

En el tablero se puede visualizar varios caracteres como:

-RC => Robot

-NÑ => Niño

-OB => Obstáculo

-SU => Suciedad

-CO => casilla de Corral

-NC => Niño capturado en un corral

-casilla en blanco, casillas no ocupadas

Modelos de Agentes considerados:

Agentes Reactivos: para este modelo además se conocía información del ambiente por lo que se pudo tomar decisiones mas inteligentes. Pero en resumen reacciona al ambiente buscando el objetivo mas cercano sea un niño o una suciedad y toma deciciones basandose en este objetivo.

Agentes de razonamiento práctico(proactivo): para este modelo el objetivo a lograr era guardar los niños en los corrales y en caso de que el corral este bloqueado tratar de limpiar. Considero que capturar a los niños sería lo más factible para la simulación dado que no se generaría más suciedad después de capturados todos

Ideas seguidas para la implementación:

1-Crear el tablero: para crear el tablero primero coloque el corral en una posición random del tablero y utilicé un algoritmo de bfs para desplegar el corral tantas casillas como fuera requerido. Luego fui colocando de forma random los demas componentes que erán requeridos (suciedad, obstaculos, niños, roboces).

2-Comportamiento de los niños: para manejar a los niños utilicé tuplas de 3 elementos (x, y, state), dos elementos para la posición en el tablero y uno para el estado del niño. Los estados para el niño son: libre, capturado por un robot y encerrado en un corral (0, 1, 2). con estos estados manejo el movimiento del niño y controlo en el momento de generar suciedad si se movió o no a traves de la antigua posición y luego de moverse (o no), en caso de no moverse no lo tengo en cuenta para generar suciedad.

3-Obstáculos: para que un niño mueva un obstaculo reviso en la dirección en q se va a mover el niño si el obstáculo se puede mover desplazandome en esa dirección por el tablero hasta encontrar un espacio vacio (en caso de que se pueda desplazar) o un espacio ocupado (en caso de no poder moverse).

4-Roboces: Para los roboces utilice dos posibles tipos, a los que llamé DumRobot y SmartRobot. el DumbRobot reacciona a lo mas cercano que tenga, sea una suciedad o un niño. En caso de caminar sobre un niño lo carga y luego empieza a buscar el corral (no deja al niño hasta llevarlo a un corral), en caso de encontrar una suciedad siempre la limpia. El SmartRobot a diferencia del anterior tiene como prioridad o meta encerrar a los niños primero para luego limpiar, de esta forma los niños no aportaran suciedad al ambiente y se lograra el objetivo. Este tipo tambien se encarga de que dos roboces no avancen hacia el mismo objetivo si existen otros objetivos a los que ir. Los roboces son representados en tuplas al igual que los niños (x, y, state) en el caso del DumbRobot y (x, y, state, target) en el caso del SmartRobot. Los posibles estados del robot son: libre, cargando un niño, sobre una suciedad, dejando un niño, sobre una casilla de corral, sobre una suciedad con un niño cargado y sobre una casillad de corral con un niño cargado. Estos estados son utilizados para en el movimiento del mismo saber a donde ir y que dejar en la posición actual después de moverse.

Los métodos para el movimiento de los roboces, procesan el movimiento de un robot a la vez, tomando en cuenta el estado del robot en el momento indicado. Si el estado es 2 (está sobre una suciedad) procede a limpiar y termina el turno de ese agente. Si el estado es 1 (tiene un niño capturado) procede a buscar el corral moviendose de 2 casillas en 2 casillas, de este movimiento se crean 2 estados más: 5 cargando un niño sobre suciedad y 6 cargando niño sobre corral. en estos casos lo que varía es el valor que se deja en la casilla al abandonar esa posición (2 o 3 respectivamente). Si el estado es 4 (esta sobre un corral vacío pero no carga un niño) avanza dejando en la posición anterior un valor correspondiente al corral en el tablero. Si el estado es 3 (dejó un niño en un corral), reemplaza el valor de esa posición al irse con un 6 (valor correspondiente a una casilla de corral ocupada por un niño). Por último si el estado es 0 se mueve normalmente dejando un 0 (valor de una casilla vacía) en la posición anterior.

5-Dejar a un niño en un corral: podía pasar que al dejar los niños en los corrales se trancara el corral y fuera inaccesible por los roboces cargando un niño, por tal decidi hacer una funcion bfs sobre el corral a partir de la posición de llegada del camino para determinar la casilla del corral mas alejada de este lugar para asi evitar hacer inaccesible el corral.

6-Para generar suciedad analizo la cuadrícula de 3x3 apartir de donde estaba el niño antes de moverse y calculo la cantidad de niños y casillas vacias de la cuadricula, luego a partir de la cantidad de niños tomo la cantidad máxima posible a generar. Luego me quedo con el mínimo entre la cantidad de casillas vacias y el máximo posible de suciedad a generar, en caso de ser 1 se genera con un 50% de probabilidad. Si es mayor que 1, utilizo un valor random entre 0 y la cantidad posible a generar y se genera aleatoriamente en las posiciones disponibles.

7-Random: utilice System.Random para generar valores random a través de un generador y una semilla, luego de utilizar una semilla utilizaba el valor resultante sin tope como siguiente semilla y devolvía además el valor módulo un parámetro para asi poder generar un valor random entre 0 y ese parámetro.

Consideraciones:

Entre las consideraciones que tomé están:

-Tomar solo 4 direcciones: este me parecio el movimiento natural a tomar, en cualquier caso es adaptable a otra cantidad de direcciones.

-La cuadricula de 3x3 para generar suciedad es tomada alrededor de la posición anterior de la que se movió el niño.

-Un niño se mueve con un 50% de probabilidad.

-La variación aleatoría del ambiente corresponde a los niños cuando ensucian.

-Si por alguna circunstancia un robot carga un niño y no existe camino hacia un corral comienza a limpiar con el niño cargado.

-Se pone una cantidad de turnos tope para parar la simulación en caso de ser demasiado larga.

Después de varias simulaciones, considero que el mejor modelo que utilicé fue el reactivo, pues este limpiaba suciedad y en caso de tener a un niño cerca lo capturaba, al contrario el otro modelo se enfocaba en guardar a los niños y en la mayoría de los casos se demoraba mucho en capturarlo, lo que demoraba más el desarrollo de la simulación. en otros casos era más factible y más rápido capturar a los niños primero pero en general el otro funcionó mejor.

Sobre todo podemos ver que en dependencia de los valores de t cambia el modelo más factible, a pequeños valores de t el reactivo es mucho más práctico pero a valores un poco más considerables de t el proactivo es más factible porque eliminan la amenaza del niño.