simulog-project

Generales del Estudiante

- Luis Enrique Dalmau Coopat C411

Aplicación

- La aplicación es un paquete del gestor de Haskell **stack**.
- Se usó Haskell *8.10.7* por problemas de compatibilidad.

Instalación

1. Instalar **stack**.
2. Descargar la versión de Haskell necesaria
    - **stack setup 8.10.7**

Ejecutar

1. Abrir consola en la carpeta raíz del proyecto
2. Ejecutar **stack run**

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 varíar 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 mue- ven, si es posible (si la casilla no está ocupada: no tiene suciedad, no está el corral, no hay un Robot de Casa), y aleatoriamente (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 aparezla suciedad. Si en una cuadrícu- la de 3 por 3 hay un solo niño, entonces, luego de que él se mueva aleato- riamente, 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 adyacentee, las que decida. Solo se mueve una casilla sino carga un niño. Si carga un niño pude 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 2niño 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 vacias no están sucias.

Principales Ideas seguidas para la solución del problema

- Se tomó a todos los elementos como agentes.
- El Tablero es una matriz de N x M y tiene una lista de elementos, ademas de llevar el turno y cada cuantos turnos ocurre el cambio aleatorio del ambiente.
- El Robot de Casa es un agente que se mueve aleatoriamente, limpia suciedad si se encuentra en una casilla con suciedad, carga niños si se encuentra a alguno en su camino y si encuentra un corral vacío con el niño cargado, lo deja en el corral.
- En otro modelo se intentara que el robot tenga metas trazadas
- Los niños se mueven aleatorio con cierta probabilidad y tambien ensucian con otra probabilidad.
- Utilizar algoritmos de recorrido para trazar metas al agente en caso de que sea el modelo 2.

Modelos de Agentes considerados (por cada agente se deben presentar dos modelos diferentes)

- Modelo 1:
  - Agentes reactivos-proactivos (el Robot de Casa)
  - Agentes puramente reactivos (los niños, la suciedad, el corral y los obstáculos)
- Modelo 2:
  - Agentes puramente reactivos (los Robot de Casa ,los niños, la suciedad, el corral y los obstáculos)

Ideas seguidas para la implementación

- Se usaron datatypes para representar al ambiente, al elemento(el cual posee un campo skill para saber si usa su habilidad o no) y al tipo de elemento(el cual puede ser niño, suciedad, corral, robot de casa, obstáculo)
- Se utiliza una funcion startSimulation la cual comienza el proceso de obtener un nuevo estado y chequear si se puede parar la simulacion o no.
- Se crean varias funciones para las distintas acciones de los elementos(Muchos de ellos no hacen nada por lo que se les asigna la funcion doNothing que no modifica el ambiente).
- Implementar BFS en Haskell para el modelo 2.
- Utilizar la funcion foldl para simular ciclos de actualizacion de estados.

- Implementar una funcion que devuelva un entero aleatorio en un rango dado.

Consideraciones obtenidas a partir de la ejecución de las simulaciones del problema (determinar a partir de la experimentación cuáles fueron los modelos de agentes que mejor funcionaron)

- El modelo 2 mejora el tiempo de ejecucion de la simulación, ya que no se generan ciclos innecesarios, lo cual mejora el rendimiento de la simulacion, pero tambien se demora un poco mas con respecto a algunas corridas del modelo 1 puesto que si cambia el ambiente es requerido reconsiderar nuevas metas para los agentes.
- El modelo 1 al ser puramente reactivo no traza metas y puede darse la situacion de que no se avance lo suficiente en ambientes muy grandes por ciclos de acciones.
- Como ventaja el modelo 1 es mucho más simple de implementar ya que no necesita de algoritmos de recorrido ni coordinar distintos agentes.
- En ambientes con poca variación de elementos, el modelo 1 es mucho más eficiente que el modelo 2, puesto que no se toma el tiempo de reconsiderar o plantear nuevas metas.