Simulación y Programación Declarativa

Proyecto de simulación de agentes en Haskell

Claudia Olavarrieta Martínez

C411

1. Problema

El ambiente en el cual intervienen los agentes es discreto y tiene la forma de un rectángulo de NxM . El ambiente es de información completa, por tanto todos los agentes conocen toda la información sobre el ambiente. 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 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 aparezca suciedad. Si en una cuadrícula de 3 por 3 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 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 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. Solución del problema

La representación del ambiente se realiza en una matriz donde cada casilla de la misma constituye una posible configuración de las posiciones entre los elementos. Las posibles casillas son:

- Empty: Casilla vacía. Representada por "| |"
- Dirt: Casilla sucia. Representada por "|D|"
- Corral: Casilla de corral. Representada por "|-|"
- Obstacle: Casilla obstáculo. Representada por "|X|"
- Kid: Niño. Representada por "|K|"
- KidInCorral: Niño en el corral. Representada por "|k|"
- Robot. Representado por "|R|"
- KidAndRobotInCorral. Niño y Robot en el Corral. Representada por "|r|"

En cuanto al Robot, este puede estar realizando distintas acciones (en el tablero todas se representan con |R|).Las acciones son:

- Moverse: El robot se mueve por casillas vacías del tablero.
- Limpiar: El robot se desplazo a una casilla sucia y procede a limpiarla
- Moviéndose por corral: El robot se desplaza por una casilla del corral.
- Moviéndose por casilla con corral y niño: *Se asume que el robot (siempre que no este cargando un niño) puede moverse por una casilla de corral donde hay un niño ya guardado.*

- Cargando niño: Puede moverse para cualquier casilla siempre que no sea un obstáculo o un niño. *Se asume que el robot puede limpiar el suelo mientras carga a un niño.*
- En un corral con un niño. Sucede cuando el robot deja al niño en el corral que le corresponde.

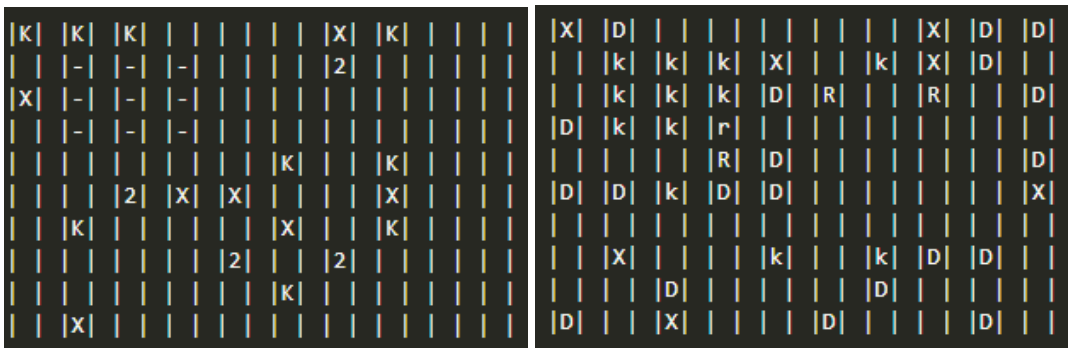

Figure 1: Ejemplo de configuración de tablero inicial (izq) y final (der).

Se implementaron dos modelos de agentes para el funcionamiento de los robots. La clasificación de estos es en Agente Reactivo y Agente Deductivo.

2.1 Agente Reactivo

Fueron programados dos tipos de robots que se corresponden a un funcionamiento totalmente reactivo.
Los **RobotDirt** están encargados únicamente de limpiar la suciedad de la casa. Sus acciones se resumen en buscar la suciedad más cercana siempre que exista un camino para llegar y limpiarla. Estos son los más básicos y se puede escoger si incluirlos en la simulación o no.
Los **RobotKid** se encargan de recoger a los niños y llevarlos para el corral. Para esto buscan al niño más cercano siempre que exista un camino válido y buscan el camino para llegar al corral correspondiente. Si se encuentran alguna suciedad mientras están caminando, ya sea para recoger al niño o en camino al corral, se detiene a limpiarlo. Una vez guardados a todos los niños en el corral el robot se dedica a limpiar la suciedad restante del tablero hasta que se cumpla que el 60% de casillas vacías estén limpias. De este tipo de robot debe haber al menos uno presente en la simulación.

2.2 Agente Deductivo

El agente deductivo posee, a diferencia de los robots antes mencionados, un estado que indica la posición a la que debe ir el robot. Para esto se siguen algunas reglas:

- Si no está llevando ningún niño se busca el niño más cercano.
  - Si la cantidad de movimientos para llegar al niño es menor que la cantidad de turnos restantes antes de que cambie el ambiente, entonces lo recoge.
  - En caso contrario, se busca la suciedad más cercana.
- Si recoge a un niño se fija como objetivo la posición libre del corral correspondiente.
- Si no quedan niños en el tablero se fija como objetivo la suciedad más cercana.

3. Resultados obtenidos

Se ejecutaron simulaciones para comparar los resultados de los dos modelos distintos que se implementaron. Para esto se empleó el mismo tablero con el objetivo de que fueran lo más similares las condiciones, las únicas posiciones que difieren son las iniciales de los robots.
A continuación se muestra una tabla que permite comparar los resultados, donde t = Cambio de Ambiente, size = Tamaño del Tablero, corral = Tamaño del Corral, obs = Cantidad de Obstáculos, RRS = Cantidad de robots reactivos de suciedad, RRN = robots reactivos de niños, RD = robots deductivos. Los turnos reactivo y deductivo indican la cantidad de turnos hasta que se completó la simulación.
Para analizar los resultados se toma, por ejemplo, los números de la primera fila 33-36-23 en la columna de turno reactivo y 27-43-47 en la columna de turnos deductivos significa que en una primera corrida el reactivo terminó en 33 turnos y el deductivo en 27, en una segunda ejecución terminó en 36 turnos el reactivo y 43 el deductivo y así sucesivamente.

t	size	corral	niños	obs	RRS	RRN	RD	Turnos Reactivo	Turnos Deductivo
3	5x5	2x2	4	5	0	2	2	33-36-23	27-43-47
30								19-22-33	13-23-76
80								12-15-20	16-38-19
10	10x10	5x5	25	10	1	2	3	218-181-195	295-348-257
50								130-169-112	195-181-141
100								154-170-117	185-204-230
10	12x15	8x8	64	35	0	6	6	388-444-375	455-554-438
50								712-609-276	536-389-405
100								909-1718-2514	1110-986-1081

Se puede observar en la primera sección de la tabla que con tamaños pequeños para el tablero, sin importar la cantidad de turnos para el cambio de ambiente los resultados fueron bastante similares.
Para la segunda sección de la tabla con tablero de 10x10 los resultados fueron mejores para los robots reactivos, habiendo poca diferencia entre la cantidad de turnos.
En la tercera sección si puede apreciarse un cambio en cuanto a la cantidad de turnos de

los robots deductivos. Se puede ver que para un intervalo de cambio de turno de 10 los robots reactivos fueron mucho mejores. Luego, cuando los turnos fueron mucho mayores el robot deductivo tuvo un mejor comportamiento como era de esperarse. Por ejemplo en el intervalo de tamaño 100 se ve su superioridad.
Puede concluirse que para un menor intervalo de cambio de ambiente los robots deductivos funcionan peor puesto que deben basar su funcionamiento en movimientos puramente reactivos, para cambios de ambiente con mayor espacio entre ellos, los robots deductivos analizan las posibilidades de éxito de sus movimientos antes de realizarlos, lo que les evita moverse innecesariamente(por ejemplo si no alcanzan al niño a tiempo). Mientras que los reactivos se mueven hacia el objetivo más cercano, ya sea que "lleguen a tiempo" o no.
Remarcar el hecho de que muchos aspectos de las simulaciones se comportan de forma aleatoria lo que puede en algunos casos beneficiar a una simulación sobre otra, por esta razón se incluyeron 3 resultados de cada tipo de ejecución para tener una idea más precisa.
Otro punto a destacar es que los agentes deductivos pueden ser mejorados agregándoles una mayor cantidad de reglas que provoquen un comportamiento más inteligente.
Otro aspecto es que el objetivo del 60% de casillas limpias no se cumple todo el tiempo, es uno de los requerimientos a conseguir para finalizar pero no necesariamente sucede durante toda la simulación. Téngase en cuenta que hay una mayor cantidad de niños que robots, que se puede dejar una gran cantidad de suciedad por turno y que el robot tarda un turno completo en limpiarlo, es por esto que la estrategia fue siempre recoger primero a los niños.

4. Consideraciones del Código

El código fue escrito en Haskell utilizando todas las prestaciones que brinda este lenguaje. Para ejecutar el programa se debe emplear algún compilador de Haskell como hugs, ghci o algún otro. Para correr una simulación se deben seguir los siguientes pasos:

1. En el archivo *Constants.hs* modificar los valores que se deseen para la ejecución.
2. Abrir una terminal en el directorio del proyecto y ejecutar el compilador que corresponda.
3. Cargar el módulo Main con el comando ‘:l Main.hs‘
4. Escribir ‘main_reactive‘ o ‘main_deductive‘ según la simulación que se desee ejecutar.

La implementación del código está dividido en distintos ficheros.

4.1 Board.hs

Contiene toda la lógica correspondiente al tablero. Las funciones que se encargan de llenar el tablero al inicio de la simulación, y las encargados de cambiar el ambiente en el turno correspondiente. En el cambio de ambiente se mueven todas las casillas que no pertenezcan al corral de forma aleatoria.

En el está la función que escoge la casilla del corral que debe ser llenada, para esto se escoge siempre la primera casilla libre para evitar que se bloquee alguna casilla del corral. También se encuentran métodos para comprobar el tipo de casilla, de acción que realiza el robot, etc.

4.2 Moves.hs

Contiene la lógica para realizar los movimientos dentro del tablero. Contiene todas las funciones para mover los niños y los obstáculos, así como las funciones que emplean los robots para desplazarse.

4.3 Random.hs

Debido a problemas que se presentaron para instalar paquetes se implementó un random propio que utiliza el tiempo actual de la pc.

4.4 Main.hs

Contiene las funciones *main_reactive* y *main_deductive* que se encarga de llamar a la función correspondiente para crear el tablero y comenzar la ejecución de la simulación. Se imprime el tablero en cada turno, el porcentaje de limpieza y el total de pasos.

4.5 Reactive.hs

Contiene la lógica correspondiente a los agentes reactivos. Posee las funciones para calcular el bfs para la distancia en la matriz, y para cambiar el tipo de acción del robot dependiendo de la casilla donde está y a la que se mueve. La función más significativa es *move_all_robot* la cual, mueve cada uno de los robots del tablero, con movimientos dobles si están cargando un niño, y esperando un turno si están limpiando.

4.6 Deductive.hs

Contiene la lógica correspondiente a los agentes deductivos. Posee las funciones para calcular el bfs para la distancia en la matriz, y para cambiar el tipo de acción del robot dependiendo de la casilla donde está y a la que se mueve. La función más significativa es *move_all_robot* la cual, mueve cada uno de los robots del tablero, además esta la función *objective_move* y *reactive_move* las cuales buscan el movimiento a realizar dependiendo de si poseen objetivo o no siguiendo las reglas antes mencionadas.

4.7 Constants.hs

Contiene los valores que se utilizan para ejecutar el código:

- change_environment: Valor de la t mencionada en la descripción del problema. Indica el número de turnos antes de que cambie el ambiente.

- n,m: Tamaño del tablero.
- corral_width y corral_height: Tamaño del corral.
- number_kids: Cantidad de niños.
- obstacles: Número de obstáculos.
- reactive_robots_dirt. Número de robots reactivos encargados de recoger la suciedad. Puede ser cero o mayor.
- reactive_robots_kids: Número de robots reactivos encargados de recoger a los niños. Mayor que cero.
- deductive_robots: Número de robots deductivos. Mayor que cero.

4.8 Dirección del Proyecto

https://github.com/ClaudiaOM/Proyecto-Haskell