Simulación de Agentes

Carlos Rafael Ortega Lezcano
Grupo C411

Orden del Problema

El ambiente en el cual intervienen los agentes es discreto y tiene la forma de un rectangulo de N x M. El ambiente es de informacion completa, por tanto todos los agentes conocen toda la informacion sobre el agente. El ambiente puede varıar 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 varie a no ser que cambie por una accion 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 variacion 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 obstaculos, suciedad, niños, el corral y los agentes que son llamados Robots de Casa. A continuacion se precisan las caracterısticas de los elementos del ambiente:

Obstaculos: estos ocupan una unica casilla en el ambiente. Ellos pueden ser movidos, empujandolos, por los niños, una unica 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 vacıas. Esta, o aparece en el estado inicial o es creada por los niños.

Corral: el corral ocupa casillas adyacentes en numero 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 este vacıa, 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 esta ocupada: no tiene suciedad, no esta el corral, no hay un Robot de Casa), y aleatoriamente (puede que no ocurra movimiento), a una de las casilla adyacentes. Si esa casilla esta ocupada por un obstaculo este es empujado por el niño, si en la direccion hay mas de un obstaculo, entonces se desplazan todos. Si el obstaculo esta en una posicion donde no puede ser empujado y el niño lo intenta, entonces el obstaculo no se mueve y el niño ocupa la misma posicion. Los niños son los responsables de que aparezla suciedad. Si en una cuadrıcula de 3 por 3 hay un solo niño, entonces, luego de que el se mueva aleatoriamente, una de las casillas de la cuadrıcula anterior que este vacıa puede haber sido ensuciada. Si hay dos niños se pueden ensuciar hasta 3. Si hay tres niños o mas pueden resultar sucias hasta 6. Los niños cuando estan 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. Tambien puede realizar las acciones de limpiar y cargar niños. Si se mueve a una casilla con suciedad, en el proximo turno puede decidir limpiar o moverse. Si se mueve a una casilla donde esta un niño, inmediatamente lo carga. En ese momento, coexisten en la casilla Robot y niño. Si se mueve a una casilla del corral que esta vacıa, 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 proximo turno, en la misma casilla, Robot y niño.

Ideas seguidas para la solución del Problema

Se definió el ambiente en el cual nuestro agente realizará sus tareas. Este se definió empleando las condiciones impuestas en el problema, el ambiente tiene dos importantes funcionalidades: **remake** y **next**. Como es necesario cada t turnos realizar una reorganización del ambiente entonces las entitades que intervienen en el ambiente son objetos los cuales mantienen un estado acorde a su interacción con otras entidades (por ejemplo un corral que tiene un niño dentro luego de una reorganización continuará teniendo la misma instancia asociada). De esta forma podemos describir de una forma más exacta las acciones del robot y los niños

en el ambiente. Para avanzar al próximo estado del ambiente se realiza next este computa los movimientos de las entidades del ambiente además de la generación de basura y el movimiento de objetos, de esta forma describimos el cambio natural del ambiente.

Contamos con dos tipos de agentes, los cuales se diferencian en la forma que interpretan el ambiente y las prioridades que tiene a la hora de lograr el objetivo. Los agentes definen 3 funciones básicas: **next**, **see** y **action**. Para ambos agentes **action** será la misma ya que describe que acciones hace de acuerdo a su interacción con las entidades. Es importante aclarar que siempre que le sea posible limpiar una casilla en la que se encuentre lo hará, no ignorará esta casilla, de esta forma siempre se intenta reducir el porciento de basura en caso que se realice otra acción. La definición para el conjunto de acciones y prioridades que da el robot.

```
Action (env):
    bot.looked ← Entidad seleccionada por el bot
    x, y ← bot.position
    if env[x, y].dirty then bot.clean
    if env[x, y].cradle and bot.carry then bot.drop
    if bot.looked is child then bot.catch_child
    if bot.looked then bot.move_to_entity
    bot.stop
```

Como podemos ver el robot primeramente intentará limpiar la casilla en la que se encuentra si esta se ensucia y termina su turno, en caso de estar cargando un niño y encontrarse en un corral deja el niño, el robot tiene un atributo **looked** el cual representa la entidad a la cual le presto atención cuando analizó el ambiente, en caso que sea un niño significa que no tiene carga por lo tanto intentará recogerlo para reducir la basura generada y tener un paso extra, si por otra parte contempla una cuna o una casilla sucia intentará llegar a esta por el camino más corto posible. Este esquema es muy parecido a la arquitectura de Brooks vista en [1].

Modelo de Agentes

Como se expuso anteriormente la diferencia en los modelos empleados radica en como el agente considera las prioridades a la hora de realizar las tareas. Para el problema se consideraron 2 modelos, ambos tienen en común que siempre que el robot llegue a una casilla sucia la limpiará en el próximo turno, el robot siempre intentará tener un niño para maximizar la cantidad de pasos que puede dar

Modelo 1: Este modelo realizará la acción más cercana posible, si tiene más cerca una cuna vacía entonces dejará al niño, si tiene una basura más cerca, se moverá a limpiarla, este modelo sigue una concepción similar a los agentes reactivos, contempla el ambiente y realiza una acción sin observar las consecuencias que tendrá sus acciones, en este caso el agente no tiene memoria por lo tanto es puro, reacciona a cada cambio del ambiente.

La función **see**, que define el comportamiento del agente acorde a lo que tiene en el ambiente es:

```
See (env):
    if not bot.carry then bot.search_child
    else near{bot.seach_cradle or bot.seach_cradle}
```

En este caso siempre intentamos cargar un niño, en caso de tenerlo buscamos lo más cercano al robot, el corral o una casilla sucia.

Modelo 2: Este modelo va más enfocado en ganar, por lo tanto su objetivo es buscar a los niños y ponerlos en el corral, este modelo no lo consideramos proactivo, ya que aunque tome la iniciativa para buscar limpiar toda la casa, no considerá que tan próximo este del despido para tomar acciones con respecto a esto.

```
See (env):
    if not bot.carry then bot.search_child
    if bot.carry then bot.search_cradle
    bot.seach_trash
```

En este caso el robot busca igual que el primero tomar un niño y pero ahora siempre lo moverá a la cuna disponible, de esta forma intentará primero recoger a todos los niños e irá limpiando la basura a medida que se mueve.

Adicionalmente se implementó otro modelo, el cual intenta recoger a todos los niños pero en caso que su acción pueda generar un aumento al 50% de basura en el ambiente, dará más prioridad a limpiar el ambiente para no ser despedido.

Es importante aclarar que el robot y los niños se mueven en las 8 direcciones, además que a primera vista se aprecia que en caso que la tarea de recoger un niño primero se dificulta excesivamente el robot siempre será despedido ya que en ambos modelos es una prioridad buscar un niño, para ello se realizó otro agente que busca igualar todas las tareas en cuanto al tipo de entidad detectada, pero continuando dando prioridad a las más cercanas.

Simulaciones Realizadas

Se generarón 10 ambientes para cada uno se hizo 30 corridas usando cada modelo de agente (los 2 principales expuestos). Al analizar los resultados podemos concluir que el segundo modelo fue mejor para resover el problema, incluso las veces que no puedo completar la tarea y fue despedido no dejo una cantidad grande de basura en el ambiente. Si observamos los ambientes generados podemos ver como existen algunos que por sus características serán muy fáciles para el robot o no habrá forma que consiga evitar ser despedido. Principalmente el ambiente 3 y 10, los cuales parecen ser bastantes complejos, podemos ver como intentar colocar todos los niños en sus corrales da mejores resultados, además el ambiente 1, presenta un 40% de basura, es

el que más basura tiene, en este el primer modelo falla todas las veces dejando en promedio 30% de basura, mientras el segundo modelo solo falla una vez.

La información de los ambientes es la siguiente:

```
Simulation Report

Environments
Done 1 -- 11x6, t=5, Trash:0.4%, Objs:9, Kids:4
Done 2 -- 5x10, t=7, Trash:0.24%, Objs:7, Kids:2
Done 3 -- 9x8, t=5, Trash:0.33%, Objs:9, Kids:4
Done 4 -- 10x11, t=5, Trash:0.32%, Objs:15, Kids:6
Done 5 -- 8x10, t=7, Trash:0.18%, Objs:8, Kids:3
Done 6 -- 8x8, t=6, Trash:0.27%, Objs:6, Kids:6
Done 7 -- 12x5, t=8, Trash:0.37%, Objs:8, Kids:4
Done 8 -- 5x11, t=8, Trash:0.19%, Objs:6, Kids:3
Done 9 -- 10x12, t=7, Trash:0.26%, Objs:12, Kids:4
Done 10 -- 8x11, t=6, Trash:0.28%, Objs:11, Kids:6
```

Los valores resultantes para el modelo 1 el cual consideramos como un agente reactivo:

```
Info for Reactive
1: {'Despedido': 30, 'Casa Limpia': 0, 'Trash': 30.5}
2: {'Despedido': 0, 'Casa Limpia': 30, 'Trash': 0.0}
3: {'Despedido': 30, 'Casa Limpia': 0, 'Trash': 35.67}
4: {'Despedido': 30, 'Casa Limpia': 0, 'Trash': 53.07}
5: {'Despedido': 0, 'Casa Limpia': 30, 'Trash': 0.0}
6: {'Despedido': 30, 'Casa Limpia': 0, 'Trash': 31.53}
7: {'Despedido': 5, 'Casa Limpia': 25, 'Trash': 4.5}
8: {'Despedido': 1, 'Casa Limpia': 29, 'Trash': 0.87}
9: {'Despedido': 29, 'Casa Limpia': 1, 'Trash': 60.63}
10: {'Despedido': 30, 'Casa Limpia': 0, 'Trash': 41.33}
```

Y los valores para el segundo modelo el cual se considera proactivo

```
Info for Proactive
1: {'Despedido': 1, 'Casa Limpia': 29, 'Trash': 1.03}
2: {'Despedido': 0, 'Casa Limpia': 30, 'Trash': 0.0}
3: {'Despedido': 26, 'Casa Limpia': 4, 'Trash': 30.03}
4: {'Despedido': 30, 'Casa Limpia': 0, 'Trash': 51.23}
5: {'Despedido': 0, 'Casa Limpia': 30, 'Trash': 0.0}
6: {'Despedido': 27, 'Casa Limpia': 3, 'Trash': 27.13}
7: {'Despedido': 6, 'Casa Limpia': 24, 'Trash': 5.67}
8: {'Despedido': 0, 'Casa Limpia': 30, 'Trash': 0.0}
9: {'Despedido': 0, 'Casa Limpia': 30, 'Trash': 0.0}
10: {'Despedido': 16, 'Casa Limpia': 14, 'Trash': 22.6}
```

Para contrastar con estos resultados tambien se implementaron otros dos modelos como se vio anteriormente. Estos resultados pueden verse si se corre la simulacion definida en el archivo `simulator.py`. El agente que busca como objetivo evitar el despido resulta muy malo para resolver el problema ya que siempre termina despedido debido a que es preferible iontentar ubicar a los niños primeramente sin tener en cuenta si esta acción implica un aumento en la basura del ambiente. Este modelo se penso para corresponderse a la representación de *goaldirect* expresada en [2], pero los cambios cada t turnos resultan perjudiciales para cualquier intento de crear una estrategia a largo plazo. El otra agente realiza la tarea más próxima que tiene, ya sea buscar a un niño, dejarlo en una cuna si es que lo carga y la tiene cerca o limpiar la basura cercana.

Repositorio

https://github.com/CRafa97/agent-simulation

References

[1] Temas de Simulación, Dr. Luciano García Garrido

[2] Artificial Intelligence a 2020, Stuart Russell, Peter Norving