Jorge Daniel Valle Díaz C-411
jorge.valle@estudiantes.matcom.uh.cu

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, o no hay un Robot de Casa), y aleatoriamente (puede que no ocurra movimiento), a una de las casillas 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 cuadricula de 3 por 3 hay un solo niño, entonces, luego de que él se mueva aleatoriamente, una de las casillas de la cuadricula 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.

Objetivos
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.
Debe programar el comportamiento del robot por cada turno así como las posibles variaciones del ambiente.

Principales Ideas seguidas para la solución del problema

En ambientes cambiantes o dinámicos, como este, se aconseja utilizar modelos de Agente Reactivo, Ellos basan su decisión enteramente en el presente, sin referencia a todo lo pasado debido a que simplemente responden directamente a los cambios que ocurren. Así, el Robot de Casa debería comportarse de forma reactiva, tomando estrategias acordes, para la solución.

Modelos de Agentes considerados

Dado que el objetivo es mantener limpio el cuarto, y que los niños son los que producen y agregan nueva basura al sistema, una primera idea sería tratar de cargar cada niño y colocarlo en el corral, y luego limpiar sin contratiempos el sistema. Así, solo estaríamos considerando los estados del agente en un inicio (si está cargando un niño lo guarda, y si no busca uno) y al final, obtenemos información del ambiente para limpiarlo.

Tomando esta idea, se puede derivar una aparentemente más eficiente, considerando que cuando sostiene un niño el robot puede aún limpiar, se puede verificar, si en la casilla en la que se encuentra, hay basura y limpiarla, así estaría obteniendo información del sistema, a cada paso, mientras cumplo la acción "principal" de llevar el niño al corral. Al terminar, se limpia la basura restante en el ambiente.

Otra forma de mirar el problema, es preocuparse de la basura, solo cuando esta se convierte de verdad, en un problema, así podemos establecer un concepto de "ambiente bastante sucio" y dedicarnos a salir de ese estado, una vez hecho esto, concentrarnos solo en guardar a los niños**.** Este concepto, se puede expresar, en términos de %, se toma como "bastante sucio", cuando la cantidad de basura en sistema sobrepasa el 40%, aquí entonces, para prevenir un posible despido, el robot limpia, hasta reducirlo, y luego continua su comportamiento, según sus estados internos.

Así, tenemos 3 formas de abordar el problema (modelos o comportamientos), sensible a la basura, de estado interno o reactivo ante el sistema.

Ideas seguidas para la implementación

Al existir similitudes en el concepto de Agente y Objeto, se programó un sistema orientado a objeto que consta de 3 clases. El ambiente es representado en la clase Enviroment, que incluye atributos para el espacio, la cantidad de niños**,** así como otros campos de información, contiene un campo referente al Robot de Casa, para que el objeto de esta clase pueda obtener su completa información. La clase Robot, cuenta con los métodos de toma de decisiones (modelos del agente Robot de casa), y métodos para indicar la acción, sus atributos son, un apuntador al ambiente, la posición en el mismo, y un atributo que denota si el robot carga o no a un niño**.** Estos últimos se reflejan en la clase Kid, que igualmente contiene métodos para generar su movimiento y la forma que tienen de crear basura en el sistema.

```python
class Enviroment:

    def __init__(self, n,m,dirt_percent,obst_percent,kids, times):
        self.world = []
        self.amount_kids = kids
        self.kids = []
        self.obs = round(((n*m)*obst_percent)/100)
        self.dirt = round(((n*m)*dirt_percent)/100)
        self.times = times
        self.dimensions = (n,m)

    def __str__(self):
        return (f'{self.amount_kids} kid(s), {self.dirt} dirty box(es)')

    def initialize(self):
        self.create_world()
        self.genBabyCradle()
        self.genObstacles()
        self.genKids()
        self.genDirt()
        self.genRobot()
```

Los objetos del sistema llamados obstáculos son generados con la condición de que no quede inconexo el ambiente, es decir, que el robot pueda llegar a cada espacio vacío en este, solo con los movimientos permitidos.

```python
def is_connected(world):...
```

La función *is_connected()* verifica la condición explicada anteriormente, se le pasa, un "mundo" o ambiente como parámetro, y devuelve verdadero si este es conexo.

Cada niño se mueve de forma aleatoria, pero que hacer cuando en su camino se encuentra obstáculos y la forma de generar basura, si cuentan con algunas reglas específicas.

```python
class Kid:

    def __init__(self,pos,env):
        self.pos = pos
        self.world = env.world
        self.env = env
        self.in_cradle = False
        self.in_robot = False

    def move(self):...

    def kids_around(self):...

    def add_dirt(self,amount_dirt,e_boxes):...
```

Las tres acciones principales que debe realizar el robot, son, recoger un niño, llevarlo a la cuna y limpiar, aquí se representan, así las tres formas de comportamiento del robot que fueron analizadas.

```
class Robot:

    def __init__(self, pos,env):

        self.carrying_kid = False
        self.pos = pos
        self.world = env.world
        self.env = env

    def pick_kids(self): ···

    def clean_room(self): ···

    def put_kid_to_bed(self): ···
```

Los 10 tableros están fijados, para variar el comportamiento, así, los que tienen las mismas dimensiones no comparten las características para lograr esta variedad. Hay tableros con un amplio por ciento de obstáculos y bajo por ciento de suciedad inicial y viceversa, otros con un alto número de niños y de unidades de tiempo, y otros con este acápite bajo.

Consideraciones obtenidas

La conclusión más evidente ya fue comentada con anterioridad, y es que los agentes con modelos proactivos o *goal-directed* no realizan un buen desempeño en sistemas tan cambiantes como el orientado.

De las tres estrategias implementadas que fueron llevadas a cabo, la que mejores resultados obtuvo, fue la que se acerca al comportamiento reactivo puro, que obtuvo la mayor cantidad de ocasiones en las que el robot no fue despedido. De un total de 300 simulaciones realizadas (30 por cada uno de los 10 ambientes), como promedio, cerca de 100, resultan positivas. En segundo lugar, la estrategia que contiene el concepto de “bastante sucio”, nótese, que en esta es importante, definir adecuadamente el margen de dicho concepto, ya que de esto depende su funcionamiento. Con el umbral de suciedad en el 40%, se logra un numero promedio de 70 a 80 ocasiones en las que se llega a un estado final satisfactorio.

Una última observación es hacia los obstáculos y cómo influyen en el tiempo de ejecución del algoritmo ya que, como se explicó anteriormente, para cada obstáculo, se revisa la conectividad del ambiente.