# Proyecto de Agentes
# Simulación

Dalianys Pérez Perera
C-411

# Índice

# 1. Principales Ideas para la solución

Para la simulación del problema planteado fue necesario representar cada una de sus componentes, tratando de que esta modelación fuese lo más cercano posible a la realidad del mismo. Por tanto, se definieron tres módulos principales: **Environment**, **Agent** y **Simulator**.

Respondiendo a las especificaciones del proyecto, se satisface que el ambiente sea discreto, de información completa y dinámico pues está sujeto a los cambios realizados por los agentes además de la variación aleatoria que ocurre cada $t$ unidades de tiempo. También la propiedad de accesibilidad del ambiente se cumple sin la necesidad de que los agentes contengan como parte de su definición a un ambiente y tampoco este último tenga a los agentes internamente.

Tanto el robot de casa como los niños constituyen agentes mostrando su capacidad ejecutiva al poder modificar el medio en que habitan. Cada uno de ellos se especializa con su conjunto de acciones particular, quedando conformada la siguiente jerarquía:

PONER FOTO

Se implementaron dos tipos de robot de casa: **ProtectRobot** el cual brinda más prioridad a guardar los niños en el corral y el robot **CleanerRobot** quien mantiene la casa lo más limpia posible y solo lleva a un niño al corral en caso de encontrarse con él.

La simulación parte de una configuración inicial del ambiente:

- $i$: cantidad de iteraciones de la simulación.
- $t$: intervalo de las variaciones aleatorias.
- $N$: cantidad de filas del ambiente.
- $M$: cantidad de columnas del ambiente.
- `dirty_porcent`: porciento de casillas sucias.
- `obst_porcent`: porciento de obstáculos a colocar.
- `num_childs` : cantidad de niños en el ambiente.
- `bot_type`: tipo de robot de casa(ProtectRobot, CleanerRobot).

Con esta información se genera un ambiente a través de la función `restart_map` la cual además de los datos anteriores recibe al robot ya con una posición aleatoria. En un inicio solo está ubicado el robot, luego se pasa a ubicar el corral garantizando siempre que sus casillas estén dispuestas consecutivamente, posteriormente se van seleccionando las casillas sucias, los obstáculos y los niños, descartando en cada paso, las casillas seleccionadas en la repartición anterior. Se asegura además que los niños no caigan dentro del corral inicialmente y que los robot no comiencen cargando a un niño.

# 2. Modelos de Agentes

Al estar en presencia de un ambiente dinámico, se decidió que los agentes fueran reactivos, ya que los eventos que ocurren en el ambiente pueden afectar los objetivos del agente o las suposiciones en las que se basa el proceso que el agente está ejecutando para lograr su objetivo. Por tanto el agente debe ser sensitivo a estos cambios. Por otro lado, ambos robots de casa en dependencia de su estado y el del ambiente pueden determinar cumplir un objetivo( encontrar un niño, llevar un niño al corral,...), es en este punto donde se manifiestan pequeños rasgos de pro-actividad.

Los niños son agentes de tipo *Child* y la única acción que realizan es moverse aleatoriamente a una de las direcciones posibles durante el turno del ambiente, una casilla a lo sumo, pues estos pueden decidir empujar a un obstáculo y que la acción no tenga efecto. Cada actuar de un niño trae consigo la genración de basura en la cuadícula del ambiente donde está contenido.

Los robots constituyen agentes con estados, pues su proceso de toma de decisión está basado en la percepción que necesitan captar del ambiente de acuerdo al estado interno actual del robot. El comportamiento de un robot es construido a partir de un número de conductas que él mismo puede asumir en dependencia de su estado. Los posibles estados de un robot son:

- CLEAN: el robot se encuentra limpiando y siempre se mueve por el ambiente hacia la casilla sucia más cercana.
- SAVE: el robot se encuentra cargando un niño y tiene el objetivo de llevarlo hacia el corral.
- FIND: el robot se encuentra de camino al niño más cercano que no esté en un corral.

Tanto *ProtectRobot* como *CleanerRobot* tienden a priorizar más un objetivo por encima de otro, por tanto tendrán lugar conductas con mayor prioridad que otras. Importante aclarar que todos los agentes tienen las siguientes funcionalidades:

- `select_direction`: selecciona la dirección a donde se moverá posteriormente para alcanzar su objetivo actual.
- `move`: se mueve en dependencia de la dirección dada. En caso de ser un robot y está cargando un niño puede moverse hasta dos casillas.
- `do_action`: actualiza el estado del robot según la percepción que captó del ambiente y del estado en que se encontraba el propio robot. Luego con esta información es que decide cuál acción ejecutar. En el caso de los niños, no hay estados y la única acción es moverse.

En las siguientes secciones se ejemplifica lo anterior con los modelos de agentes implementados.

### 2.1. Agente ProtectRobot

Como ya se mencionó, este robot prioriza llevar todos los niños al corral desde un inicio y cuando lo logre es que se mantiene limpiando la casa. Por defecto el estado de este robot es FIND, pues cuando comienza la simulación su primer objetivo es encontrar al niño más cercano.

## 3. Ideas seguidas para la implementación

## 4. Simulaciones

### 4.1. Conclusiones